Self Assessment Report

BNU

Beaconhouse National University

School of Computer & Information Technology

B.Sc. Hons Software Engineering

Prepared by: Program Team of SCIT
Prepared by: Quality Assurance Department

Table of Contents

Executive Summary

This report is prepared at the end of the assessment cycle of School of Computer and Information Technology (SCIT-eve) of Beaconhouse National University (BNU), as per requirement of Higher Education Commission (HEC). Quality Assurance Department (QA) was formed in BNU in September 2005. Program Team Members notified by University worked with General Manager Quality Assurance to pursue the application of Self Assessment Manual in their respective department.

In School of Computer and Information Technology (SCIT-eve), B.Sc. in Software Engineering program was selected for the self assessment, evaluation and improvements. Commitment of respected Vice Chancellor to support Quality Assurance Department made the difference and resultantly, a cycle of assessment is about to complete.

Objectives

Following are the two main objectives of the self assessment report:

1. To implement Self Assessment Manual in selected program with a view to improve quality in higher education.
2. To identify the areas requiring improvements in order to achieve objectives through desired outcomes.

Execution

A soft and hard copy of self assessment manual was given to Dean and faculty. Quality Awareness presentation of Self Assessment Report (SAR) was arranged for the Dean and Program Team Members (PT) of the selected program. Hard copies of HEC issued 10 proformas with manual with 8 criterion and 31 standards were provided to PT members to evaluate their program against defined standards. The PT members with an intimate support and follow up of QA, completed the SAR and forwarded to QA.

After reviewing SAR, QA arranged visit of Assessment Team to the selected program on 2nd March, 2015. GM (QA) accompanied the AT and participated in discussions with Dean and PT members and available faculty members. Date for exit meeting was fixed as 29th April, 2015.

The implementation plan based on discussions in exit meeting have been made by In-charge Programs. They prepared it under following headings:

a. Assessment Team finding
b. Corrective Actions required
c. Resources Needed

The implementation plan indicates the resources to improve the infrastructure, environment in the classes and Laboratory manuals. The recommended target dates to complete the tasks observed by Assessment Team, presented in exit meeting on 29th April, 2015.
and approved by Vice Chancellor have been indicated in the implementation plan.

At the completion of Self Assessment cycle, QA submitted the hard and soft copy of SAR to HEC on July 10, 2015.

General Manger (QA)

Introduction

School of Computer and Information Technology, Beaconhouse National University is conscious of the fast changing market requirements, which emanate from new systems and technologies. In particular, the Internet and mobile technologies have completely changed the working of the business and commercial world.

The School introduced new programs and updated exiting ones in response to these changing needs.

The teaching – learning model followed by School of Computer& Information Technology greatly emphasizes practical work to enforce understanding of theoretical concepts.

Criterion 1: Program Mission, Objectives and Outcomes

Institution Mission Statement

"A truly national higher-education institution, emerging as a world-class Liberal Arts university with a merit-driven, need-based recruitment and admission policy at all levels; offering modern curricula in a range of conventional and new disciplines; while preserving the history and culture of Pakistani society; enriching the overall intellectual growth of a student through interaction and professional excellence."

Standard 1-1 The program must have documented measurable objectives that support institution mission statements.

Department Vision Statement (School of Computer & IT)

To become the leading national school in business computing and software engineering education and research.

Department Mission Statement (School of Computer & IT)

Our mission is to build a successful career for our students. We will provide them with a high quality, enjoyable learning experience and transform them into managers who are technically superb, socially responsible and professionally accomplished

We will achieve our mission by offering market relevant academic programs in IT, Business and Management in a progressive and friendly learning environment. We will retain highly qualified and dedicated faculty, provide up-to-date resources and pursue policies which are based on performance and merit.

Program Mission Statement B.Sc (Hons) in Software Engineering)

The mission of B.Sc. (Hons) in Software Engineering program is to prepare graduates for careers in the Information and Communication Technologies (ICT) sector and post-graduate study and research in ICT and related disciplines.

Program Educational Objectives

The B.Sc. (Hons) in Software Engineering is designed to achieve the following objectives:

1. To prepare students for software engineering positions in the Information and Communication Technology (ICT) sector of the industry.

2. To prepare students for higher education in software engineering and relevant disciplines.

3. To impart technical skills in the art and science of software development.

4. To prepare students for research and development (R&D) in software engineering and related areas.

5. To inculcate professional and ethical values in the students.

6. To develop good interpersonal and communication skills in the students, especially with relevance to their program of studies.

The School of IT has built up its academic environment keeping in view the above program objectives. The selection of faculty, design of curriculum and syllabus, instructional procedures and practice-oriented teaching help to enforce the above program objectives.

The School of IT is supported in its efforts by the Quality Assurance Department of Beaconhouse National University.

Strategic Plan

One of the goals of the School of IT is to formulate a quality assured curriculum in which various quality parameters are verifiable and bench marked.

To this end, the School of IT follows the systems and procedures prescribed by the HEC. Further, the School of IT has updated its curriculum in line with the recommendations of NCEAC, the accreditation authority set up by the HEC.

Program Objective's Assessment

The following table shows how each of the above program objectives is measured and the actions taken as a result of these measurements.

The three tools for assessments of program objectives are:

1. Employer Survey
2. Alumni Survey
3. Graduating Students Survey

Objectives	How Measured	When Measured	Improvement Identified	Improvement Made
1	Graduating Students Survey	At conclusion of four year program	On basis of final project performance	Converted some teaching classes into Lab
2	Alumni Survey	Within one year of graduation	Nil	Nil
3	Employer Survey	Within one year of graduation	Better technical writing skills required	Review of the contents of Communication Skills courses Close supervision

				of final project documents
4	Graduating Students Survey	Conclusion of four year program	Better analytical and research skills	Monitoring of final year projects with reference to the curriculum
5	a. Alumni Survey b. Graduating Students Survey	Within one year of graduation	More adherence to professional values	Case based curriculum of professional elective course
6	c. Alumni Survey d. Graduating Students Survey	Within one year of graduation	Better interpersonal skills required	Encourage students to work in teams

Table 4.1: Program Objectives Assessment

Standard 1-2: The program must have documented outcomes for graduating students. It must be demonstrated that the outcomes support the program objectives and that graduating students are capable of performing these outcomes.

Program Outcomes

B.Sc. (Hons) in Software Engineering has the following program outcomes. By the end of the program the students should be able to

1. Comprehend a problem and then propose a computer based solution for it.

2. Setup a software development environment and work in it.

3. Code, debug and test the software.

4. Analyze a problem and then represent the analysis in a prescribed document format.

5. Implement the technical, human and ethical aspects of the software solutions proposed.

6. Identify errors in the software development and then suggest measures to rectify these errors.

7. Communicate the issues and problems related to software development in a professional and readily understandable format.

8. Propose and implement end-to-end solutions.

9. Come up with software solutions which are relevant to local needs and conditions.

10. Be able to study and retrieve relevant information from software literature and research engineering publications.

Program Objectives	Program Outcomes									
	1	2	3	4	5	6	7	8	9	10
1	X	x	X	X	x	X	x	X	x	
2			X					X		x
3			X		x			X		
4								X		x
5					x		x			
6				X	x		x			

Table 4.2: Outcomes versus objectives

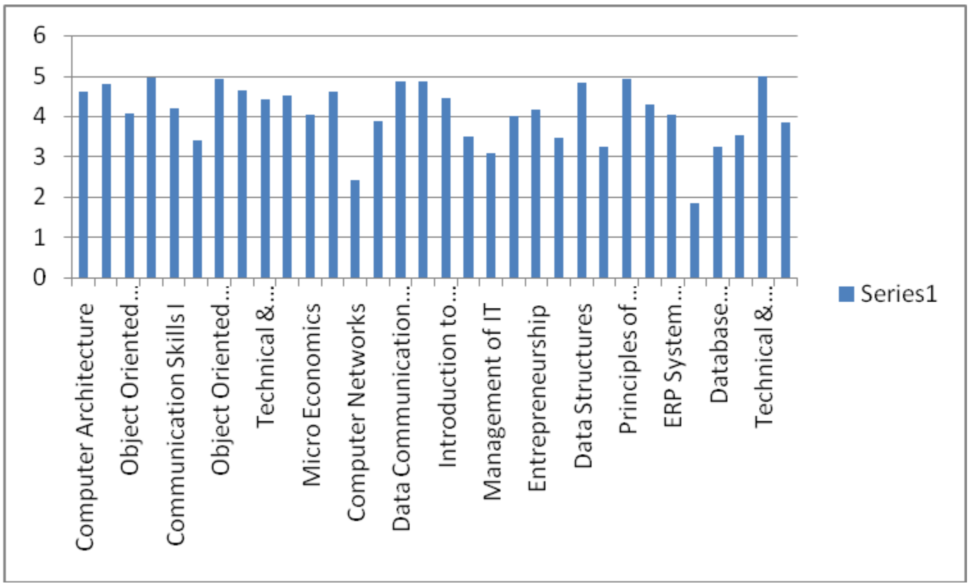
Standard 1-3: The results of the program's assessment and the extent to which they are used to improve the program must be documented.

The program assessment has been done by launching HEC Proforma number 1 and 10. The students of the program evaluated the courses and teachers of the software engineers program.

Course-wise

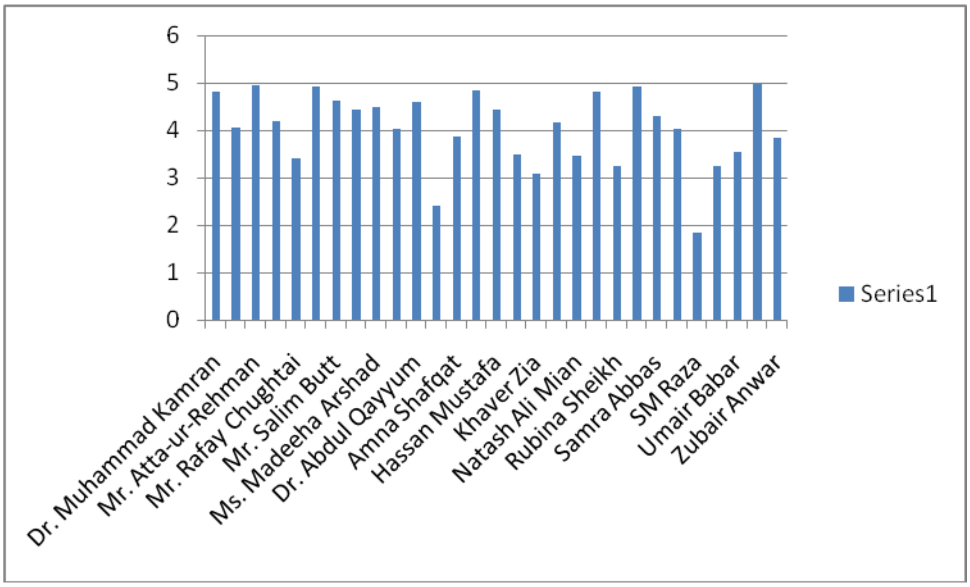
Sr. No.	Course Code	Course Title	Course Cr. Hrs.	Evaluation
1	CSC-302	Computer Architecture	3	**4.61**
2	CSC-104	Digital Logic Design	4	**4.80**
3	CSC-213	Object Oriented Programming	4	**4.05**
4	HUM-201	Islamic Studies	3	**4.94**
5	SE-101-A	Communication Skills I	3	**4.19**
6	CSC-205	Computer Organization and Assembly Language	4	**3.40**
7	CSC-313	Object Oriented Analysis and Design	3	**4.93**
8	MTH-105	Applied Physics	3	**4.62**
9	HUM-302	Technical & Professional Communication	3	**4.42**
10	MGT-305	Operations Management	3	**4.50**
11	BUS-101	Micro Economics	3	**4.03**
12	MTH-104	Business Statistics	3	**4.60**

13	CSC-203	Computer Networks	3	**2.40**
14	SE-101-F	Foundation English	3	**3.86**
15	CSC-210	Data Communication and Networks	3	**4.85**
16	CSC-314	Study of Algorithms	3	**4.86**
17	CSC-108	Introduction to Programming	4	**4.43**
18	CSC-305	Human Computer Interaction	3	**3.50**
19	MGT-303	Management of IT	4	**3.08**
20	BUS-303	Business Law	3	**4.00**
21	MGT-304	Entrepreneurship	3	**4.15**
22	CSC-107	Fundamental of Computers	4	**3.46**
23	CSC-204	Data Structures	4	**4.82**
24	SE-101-B	Communication Skills II	3	**3.25**
25	MGT-101	Principles of Management	4	**4.92**
26	MTH-101	Calculus & Analytical Geometry I	3	**4.29**
27	CSC-601	ERP System Implementation	3	**4.04**
28	CSC-105	Discrete Structures	3	**1.84**
29	CSC-211	Database Management Systems	4	**3.24**
30	CSC-110	Business Computing	4	**3.53**
31	HUM-302	Technical & Professional Communication	3	**4.98**
32	BUS-503	Business Process Modeling	3	**3.83**



Teacher-wise

Sr. No.	Name of Faculty	Evaluation
1	Dr. Muhammad Kamran	**4.80**
2	Mr. Umair Babar	**4.05**
3	Mr. Atta-ur-Rehman	**4.94**
4	Ms. Amna Shafqat	**4.19**
5	Mr. Rafay Chughtai	**3.40**
6	Mr. Nouman Ali Shah	**4.93**
7	Mr. Salim Butt	**4.62**
8	Ms. Uzma Tauqeer	**4.42**
9	Ms. Madeeha Arshad	**4.50**
10	Ms. Kiren Kabir	**4.03**
11	Dr. Abdul Qayyum	**4.60**
12	Mr. Waleed Baig	**2.40**
13	Amna Shafqat	**3.86**
14	Dr Muhammad Kamran	**4.85**
16	Hassan Mustafa	**4.43**
17	Huda Sarfraz	**3.50**
18	Khaver Zia	**3.08**
20	Mehr un Nisa	**4.15**
21	Natash Ali Mian	**3.46**
22	Nouman Ali	**4.82**
23	Rubina Sheikh	**3.25**
24	Saad Saleem Malik	**4.92**
25	Samra Abbas	**4.29**
26	Sh. Muhammad Ali	**4.04**
27	SM Raza	**1.84**
28	Toqir Ahmad Rana	**3.24**
29	Umair Babar	**3.53**
30	Uzma Touqir	**4.98**
31	Zubair Anwar	**3.83**



Standard 1-4: The department must assess its overall performance periodically using quantifiable measures.

Status Report of Students (SCIT)							
Year	**Program**	**Applied**	**Admitted**	**Left**	**Terminated**	**Graduated**	**Studying**
2004	MMS	20	15	9	-	6	0
Sub-Total		**20**	**15**	**9**	**0**	**6**	**0**
2005	MMS	10	3	0	-	3	0
	TCS	20	14	8	-	6	0
Sub-Total		**30**	**17**	**8**	**0**	**9**	**0**
2006	MMS	16	4	1	-	3	0
	TCS	25	9	8	-	1	0
	MBC	81	33	18	-	15	0
Sub-Total		**122**	**46**	**27**	**0**	**19**	**0**
2007	MBC	98	29	17	-	-	12
	FBC	63	19	16	-	-	3
	SE	41	12	8	-	-	4
Sub-Total		**202**	**60**	**41**	**0**	**0**	**19**
2008	MBC	111	40	13	3	-	24
	FBC	69	26	6	2	-	18
	SE	38	18	4	2	-	12
Sub-Total		**218**	**84**	**23**	**7**	**0**	**54**
2009	MBC	103	28	6	8	-	14
	FBC	68	25	8	7	-	10
	SE	56	18	4	0	-	14
Sub-Total		**227**	**71**	**18**	**15**	**0**	**38**
2010	MBC	101	40	7	4	15	29

	FBC	72	27	3	4	-	19
	SE	88	34	5	2	-	28
Sub-Total		**261**	**101**	**15**	**10**	**0**	**76**
2010 Evening	MBC	26	11	8	1	-	2
	SE	15	6	2	-	-	4
Sub-Total		**41**	**17**	**10**	**1**	**0**	**6**
2011	MBC	107	24	4	-	11	20
	FBC	40	9	-	-	-	9
	SE	122	30	-	-	2	30
Sub-Total		**269**	**63**	**4**	**0**	**0**	**59**
2011 Evening	MBC	18	13	-	-	-	13
	SE	31	20	-	1	-	19
Sub-Total		**49**	**33**	**0**	**1**	**0**	**32**
2012	SE	111	58		11	7	47
	MBC	92	39		5	4	34
	FBC	47	17		4	7	13
Sub Total		**250**	**114**	**0**	**20**	**0**	**94**
2012 Evening	MBC	18	3	0	2	-	1
	SE	32	15	0	2	-	13
Sub-Total		**50**	**18**		**4**	**0**	**14**
2013	MBC	105	29	4	6	22	19
	FBC	46	10	2	1	11	7
	SE	155	48	5	1	11	42
Sub-Total		**306**	**87**	**11**	**8**	**0**	**68**
2013 Evening	MBC	23	-	-	-	-	-
	SE	32	-	-	-	-	-
	MS BIT	16	-	-	-	-	-
Sub Total		**71**	**0**	**0**	**0**	**0**	**0**
2014	MBC	97	49	4	2	14	43
	FBC	29	15	-	2	10	13
	SE	126	51	4	1	11	46
Sub Total		**252**	**115**	**8**	**5**	**0**	**102**
2014 Evening	MBC	8	-	-	-	-	-
	SE	59	40	-	-	-	40
	MS BIT	2	-	-	-	-	-
Sub Total		**69**	**40**	**0**	**0**	**0**	**40**

CRITERION 2: CURRICULUM DESIGN AND ORGANIZATION

Standard 2-1: The curriculum must be consistent and supports the program's documented objectives.

Title of Degree Program
B.Sc. (Hons) in Software Engineering

Definition of credit hour:
One credit hour is 1 hour of theory lecture or 3 hours of laboratory work in a week.

Degree plan
Following is the list of all courses offered by the School of IT. Courses which are part of the B.Sc. (Hons) in Software Engineering are indicated with (*).

The matrix of the B.Sc. (Hons) in Software Engineering is given in which pre-requisite of each course is also indicated.

BASIC STREAM	S. No.	COURSE CODE	COURSE TITLE	CREDIT HOURS
Business (BUS)	1	BUS - 101	Micro Economics	3
	2	BUS - 201	Businesses & Organizations	3
	3	BUS - 202	Macro Economics	3
	4	BUS - 301	Marketing and Business Development	4
	5	BUS - 302	Management of Internet Business	4
	6	BUS - 303	Business Law	3
	7	BUS - 304	Principles of Marketing	3
	8	BUS - 4xx	Elective (Business)	3
Computers (CSC)	9	CSC - 104	Digital Logic Design (*)	4
	10	CSC - 105	Discrete Structures (*)	3
	11	CSC - 107	Fundamentals of Computers (*)	4
	12	CSC - 108	Intro. to Programming (*)	4
	13	CSC - 109	Creating Digital Content (*)	4
	14	CSC - 203	Computer Networks	3
	15	CSC - 204	Data Structures (*)	4
	16	CSC - 205	Computer Org. & Assembly (*)	4
	17	CSC - 206	Event-Driven Programming (*)	4
	18	CSC - 210	Data Communication and Networks (*)	3
	19	CSC - 211	Database Management Systems (*)	4

	20	CSC - 213	Object Oriented Programming (*)	4
	21	CSC - 302	Computer Architecture (*)	3
	22	CSC - 303	Software Engineering (*)	3
	23	CSC - 306	Computer Graphics (*)	4
	24	CSC - 307	Artificial Intelligence (*)	3
	25	CSC - 310	ERP Systems	3
BASIC STREAM		**COURSE CODE**	**COURSE TITLE**	**CREDIT HOURS**
	26	CSC - 313	Object Oriented Analysis and Design (*)	3
	27	CSC - 314	Study of Algorithms (*)	3
	28	CSC - 315	Lab Projects Course (*)	3
	29	CSC - 316	Software Construction (*)	4
	30	CSC - 317	Software Requirements Engineering (*)	4
	31	CSC - 402	Theory Of Automata	3
	32	CSC - 403	Operating Systems (*)	3
	33	CSC - 406	Internet Programming	4
	34	CSC - 408	Compiler Theory	3
	35	CSC - 410	Data Warehousing	4
	36	CSC - 411	Software Quality Engineering (*)	3
	37	CSC - 412	Software Project Management (*)	3
	38	CSC - 413	Formal Methods in Software Engineering (*)	3
	39	CSC - 414	Software Design and Architecture (*)	3
	40	CSC - 4xx	Elective (Development)	3
Information Systems (CIS)	41	CIS - 204	IS Hardware and System Software	3
	42	CIS - 301	Information Management	3
	43	CIS - 305	Human Computer Interaction	3
	44	FIN - 101	Intro to Finance	3
	45	FIN - 102	Accounting I (*)	3
	46	FIN - 201	Accounting II	3
Finance (FIN)	47	FIN - 202	Financial Management	4
	48	FIN - 203	Management Accounting	3
	49	FIN - 402	Financial Markets and Institutions	3
	50	FIN - 403	Financial Management for MBC	3
	51	FIN – 4xx	Elective (Finance)	3

BASIC STREAM		COURSE CODE	COURSE TITLE	CREDIT HOURS
Humanities (HUM)	52	HUM - xxx	Humanities (Elective)	3
	53	HUM - 3xx	Foreign Language (*)	3
	54	HUM - 201	Islamic Studies (*)	3
	55	HUM - 202	Pakistan Studies (*)	3
	56	HUM - 301	Technical & Professional Communication (*)	3
	57	HUM - 302	Psychology	3
	58	HUM - 401	Research & Professional Issues (*)	3
	59	HUM - 402	Professional Ethics (*)	3
	60	MC B 118	Media & Ethics	3
		SE 101-A	Communication Skills I (*)	3
	61	SE 101-B	Communication Skills II (*)	3
Management (MGT)	62	MGT - 101	Principles of Management (*)	3
	63	MGT - 201	Management Information Systems	3
	64	MGT - 202	Total Quality Management	3
	65	MGT - 203	Organizational Behavior	3
	66	MGT - 301	Project Management	4
	67	MGT - 302	Human Resource Management	3
	68	MGT - 303	Management of IT	4
	69	MGT - 304	Entrepreneurship	3
	70	MGT - 305	Operations Management	3
	71	MGT - 306	Leadership Skills	3
	72	MGT - 401	Business Policy	4
	73	MGT - 402	Strategic Management	3
	74	MGT - 4xx	Elective (Management)	3
	75	MTH - 101	Calculus & Analytical Geometry I (*)	3
	76	MTH - 102	Business Mathematics	3
	77	MTH - 103	Calculus & Analytical Geometry II (*)	3
Mathematics	78	MTH - 104	Business Statistics	3

(MTH)	79	MTH - 105	Applied Physics (*)	3
	80	MTH - 106	Pre-Calculus	3
	81	MTH - 201	Probability & Statistics (*)	3
	82	MTH - 202	Linear Algebra & Differential Equations (*)	3
	83	MTH - 301	Statistical Inference	3
Project (PRJ)	84	PRJ - 401	Project I (*)	3
	85	PRJ - 402	Project II (*)	3

Software Engineering Course Plan

Semester - I First Year

Course Code	Course Title	Credits
CSC - 107	Fundamentals of Computers	4
SE -101-A	Communication Skills I	3
MTH - 101	Calculus & Analytical Geometry I	3
	Total	10

Semester - II First Year

Course Code	Course Title	Credits
MMS - 101	Creating Digital Content	4
EC-108	Introduction to Accounting	3
	Total	7

Semester - III First Year

Course Code	Course Title	Credits
SE -101- B	Communication Skills II	3
MTH - 105	Applied Physics	3
CSC - 104	Digital Logic Design	4
	Total	10

Semester - IV Second Year

Course Code	Course Title	Credits
CSC - 108	Introduction to Programming	4
HUM - 203	Pakistan and Islamic Studies	3
CSC - 105	Discrete Structures	3
	Total	10

Semester - V Second Year

Course Code	Course Title	Credits
MTH - 103	Calculus & Analytical Geometry II	3
MGT - 101	Principles of Management	3

Total 6

Semester - VI Second Year

Course Code	Course Title	Credits
CSC - 213	Object Oriented programming	4
CSC - 205	Computer Org. & Assembly	4
CSC - 212	Elective (IT)	3

Total 11

Semester - VII Third Year

Course Code	Course Title	Credits
CSC - 204	Data Structures	4
CSC - 211	Database Management System	4
CSC - 210	Data Communication & Networks	3

Total 11

Semester - VIII Third Year

Course Code	Course Title	Credits
MTH - 201	Probability & Statistics	3
MTH - 202	Liear Algebra & Differential Equations	3

Total 6

Semester - IX Third Year

Course Code	Course Title	Credits
CSC - 302	Computer Archetechure	3
CSC - 313	Object Oriented Analysis & Design	3
CIS - 305	Human Computer Interaction	3

Total 9

Semester - X Fourth Year

Course Code	Course Title	Credits
CSC - 316	Software Construction	4
CSC - 314	Study of Algorithms	3
CSC - 409	Data Mining & Warehousing	3

Total 10

Semester - XI Fourth Year

Course Code	Course Title	Credits
CSC – 317	Software Requirements Engineering	4
CSC – 412	Software Project Management	3

Total 7

Semester - XII Fourth Year

Course Code	Course Title	Credits
CSC – 406	Internet Programming	4
CSC – 315	Lab Projects Course	3
CSC – 403	Operating Systems	3

Total 10

Semester - X Fifth Year

Course Code	Course Title	Credits
HUM - 401	Research & Professional Issues	3
CSC – 413	Formal Methods in Software Engineering	3
PRJ – 401	Project I	3

Total 9

Semester - XI Fifth Year

Course Code	Course Title	Credits
HUM - 4xx	Elective (Humanities)	3
HUM - 402	Professional Ethics	3

Total 6

Semester - XII Fifth Year

Course Code	Course Title	Credits
CSC – 414	Software Design and Architecture	3
CSC – 411	Software Quality Engineering	3
PRJ – 402	Project II	3

Total 9

Total Credit Hours in the Program **131**

Table 1.9: Software Engineering Course Plan

Comparison of B.Sc. (Hons) in Software Engineering Curriculum with HEC Curriculum

(A) COMPUTING COURSES (REQUIRED)

Sr No.	HEC Curriculum	Cr Hr		Sr. No.	SCIT Curriculum	Cr Hr.
1	Intro to Computing	4		1	Fundamental of Computers	4
2	Programming Fundamentals	4		2	Intro to Programming	4
3	Object Oriented Programming	3		3	Object Oriented Programming	4
4	Discrete Structures	3		4	Discrete Structures	3
5	Data Structure and Algorithms	3		5	Data Structures	4
6	Digital Logic and Design	3		6	Digital Logic Design	4
7	Operating Systems	4		7	Operating Systems	3
8	Intro to Database Systems	4		8	Database Management Systems	4
9	Introduction to Software Engineering	3		9		

10	Computer Communication and Networks	3		10	Data Communication and Networks	3
11	Human Computer Interation	3		11	Human Computer Interaction	3
12	Senior Design Project	6		12	Project (l & ll)	6
	Sub Total	**43**			**Sub Total**	**42**

(B) COMPUTING SUPPORTING COURSES (REQUIRED)

Sr No.	**HEC Curriculum**	**Cr Hr**		**Sr. No.**	**SCIT Curriculum**	**Cr Hr.**
1	Calculus and Analytical Geometry	3		1	Calculus and Analytical Geometry-I	3
2	Probability and Statistics	3		2	Probability and Statistics	3
3	Linear Algebra	3		3	Linear Algebra & Differenial Equations	3
4	Electromagnetism	3		4	Applied Physics	3
	Sub Total	**12**			**Sub Total**	**12**

(C) COMPUTING GENERAL EDUCATION COURSES (REQUIRED)

Sr No.	**HEC Curriculum**	**Cr Hr**		**Sr. No.**	**SCIT Curriculum**	**Cr Hr.**
1	English-I (Functional English)	3		1	Communication Skills l	3
2	English-II (Technical and Report Writing)	3		2		
3	English-III (Communication Skills)	3		3	Communication Skills ll	3
4	Islamic and Pakistan Studies	3		4	Pakistan and Islamic Studies	3
5	Professional Practices	3	3	5	Research and Professional Issues	3
	Sub Total	**15**			**Sub Total**	**12**

(D) SOFTWARE ENGG COURSES (REQUIRED)

Sr No.	**HEC Curriculum**	**Cr Hr**		**Sr. No.**	**SCIT Curriculum**	**Cr Hr.**
1	Software Construction	3		1	Software Construction	4
2	Software Requirements Engineering	3		2	Software Requirements Engineering	3
3	Software Design and Architecture	3		3	Software Design and Architecture	3
4	Software Quality Engineering	3		4	Software Quality Engineering	3
5	Software Project Management	3		5	Software Project Management	3
6	Formal Methods in Software Engineering	3		6	Formal Methods in Software Engineering	3
	Sub Total	**18**			**Sub Total**	**19**

(E) COMPUTING AND SOFTWARE ENGINEERING COURSES (ELECTIVE)

Sr No.	**HEC Curriculum**	**Cr Hr**		**Sr. No.**	**SCIT Curriculum**	**Cr Hr.**
1	Data Mining and Warehousing	3		1	Data Mining and Warehousing	3
2	Analysis of Algorithms	3		2	Study of Algorithms	3
3	Web Engineering	3		3	Web Engineering	4
4	Software Testing	3		4	Computer Organization and Assembly	4
5	Design Patterns	3		5	Object Oriented Analysis and Design	3
6	Artificial Intelligence	3		5	Lab Projects Course	3
	Sub Total	**18**			**Sub Total**	**20**

(F) SOFTWARE ENGINEERING DOMAIN SPECIFIC (ELECTIVE)

Sr No.	**HEC Curriculum**	**Cr Hr**		**Sr. No.**	**SCIT Curriculum**	**Cr Hr.**
1	Web Based Technologies	3		1	Creating Digital Content	4
2	Embedded Systems	3		2	Computer Architecture	3
	Sub Total	**6**			**Sub Total**	**7**

(G) COMPUTING SUPPORTING COURSE (ELECTIVE)

Sr No.	HEC Curriculum	Cr Hr		Sr. No.	SCIT Curriculum	Cr Hr.
1	Advanced Calculus	3		1	Calculus and Analytical Geometry-II	3
2	Operations Research	3		2	Graphic Design Tools	3
3	Natural Language Processing	3		3	Elective (IT)	3
	Sub Total	**9**			**Sub Total**	**9**

(H) GENERAL EDUCATION ELECTIVES (UNIVERSITY ELECTIVES)

Sr No.	HEC Curriculum	Cr Hr		Sr. No.	SCIT Curriculum	Cr Hr.
1	Accounting and Finance	3		1	Intro to Accounting	3
2	Principles of Management	3		2	Principles of Management	3
3	Psychlogy	3		3	Professional Ethics	3
	Sub Total	**9**			**Sub Total**	**9**
	Grand Total	**130**			**Grand Total**	**130**

Curriculum Breakdown

Semester	**Course Table 1.10**	**Category (Credit Hours)**				
		Math and Basic Science		**Core Courses**	**Humanities and Social Sciences**	**Technical Electives / Others**
		Math	**Basic Science**			
1	**(A) Computing Courses (Required)**			✓		✓
2	**(B) Computing Supporting Courses (Required)**	✓	✓	✓		
3	**(C) Computing General Education Courses (Required)**			✓	✓	
4	**(D) Software Engineering Courses (Required)**			✓		✓
5	**(E) Software Engineering Courses (Elective)**					✓
6	**(F) Software Engineering Domain Specific (Elective)**					✓
7	**(G) Computing Supporting Courses (Elective)**	✓		✓		✓
8	**(H) General Education Electives (University Electives)**			✓	✓	✓

Table 4.3: Curriculum Course Requirements

Courses/ Group of courses	Objectives					
	1	2	3	4	5	6
(A) Computing Courses (Required)	✓	✓		✓		
(B) Computing Supporting Courses (Required)						
(C) Computing General Education Courses (Required)				✓	✓	✓
(D) Software Engineering Courses (Required)	✓	✓	✓	✓		
(E) Software Engineering Courses (Elective)	✓	✓		✓		
(F) Software Engineering Domain Specific (Elective)	✓	✓	✓			
(G) Computing Supporting Courses (Elective)		✓	✓			
(H) General Education (University Electives)			✓		✓	✓

Table 4.4: Courses versus Program Outcomes

Standard 2-2: Theoretical background, problems analysis and solution design must be stressed within the program's core material.

Indicate which courses contain a significant portion (more than 30%) of the elements in standard 2-2.

Elements	Course (ref: Table 1.11)
Theoretical background	Group A , B, C,D,G,E,H
Problem analysis	Group B, D , E, H
Solution design	Group A,F,G,E

Table 4.5: Standard 2-2 requirement

Standard 2-3: The curriculum must satisfy the core requirements for the program, as specified by the respective accreditation body.

- ***Please refer to Standard 2- 1 (Table 1.10)***

Standard 2-4: The curriculum must satisfy the major requirements for the program as specified by HEC, the respective accreditation body / councils.

- ***Please refer to Standard 2- 1 (Table 1.10)***

Standard 2-5: The curriculum must satisfy general education, arts, and professional and other discipline requirements for the program, as specified by the respective accreditation body / council.

- ***Please refer to Standard 2- 1 (Table 1.10)***

Standard 2-6: Information technology component of the curriculum must be integrated throughout the program.

- ***Please refer to Standard 2- 1 (Table 1.9)***

Standard 2-7: Oral and written communication skills of the student must be developed and applied in the program.

- ***Please refer to Standard 2- 1 (Table 1.10)***

CRITERION 3: LABORATORY AND COMPUTING FACILITIES

The School of Computer and Information Technology contain the following Labs facility for the evening program:

There are two labs available for Software Engineering evening program

1. **Lab 1:**
   This lab contains twenty eighteen workstations. All required software for the B.Sc. (Hons) in Software Engineering program are available on each workstation.

2. **Lab 2:**
   The lab contains twenty thirteen workstations. All the workstations have the required software for the B.Sc. (Hons) in Software Engineering program.

Standard 3-1: Laboratory manuals/ documentation/ instructions for experiments must be available and readily accessible to faculty and students.

The above mentioned labs facilitate the students in performing lab exercises and projects relating to the course offered in Software Engineering program.

The following courses have lab exercises which are conducted in the above mentioned labs:

1.	ERP Systems	(3 hours per week)
2.	Software Requirement Engineering	(3 hours per week)
3.	Object Oriented Programming	(3 hours per week)
4.	Computer Organization & Assembly Language	(3 hours per week)
5.	Digital Logic Design	(3 hours per week)
6.	Software Construction	(3 hours per week)
7.	Database Management System	(3 hours per week)
8.	Data Structures	(3 hours per week)
9.	Introduction to Programming	(3 hours per week)
10.	Fundamentals of Computers	(3 hours per week)
11.	Project I	(9 hours per week)
12.	Project II	(9 hours per week)

Sample of guided lab instructions are provided in the table below:

Computer Organization & Assembly Language (CSC-205)
Fall 2014

Lab Manual

Course Instructor: Huda Sarfraz

School of Computer and Information Technology
Beaconhouse National University

Table of Contents

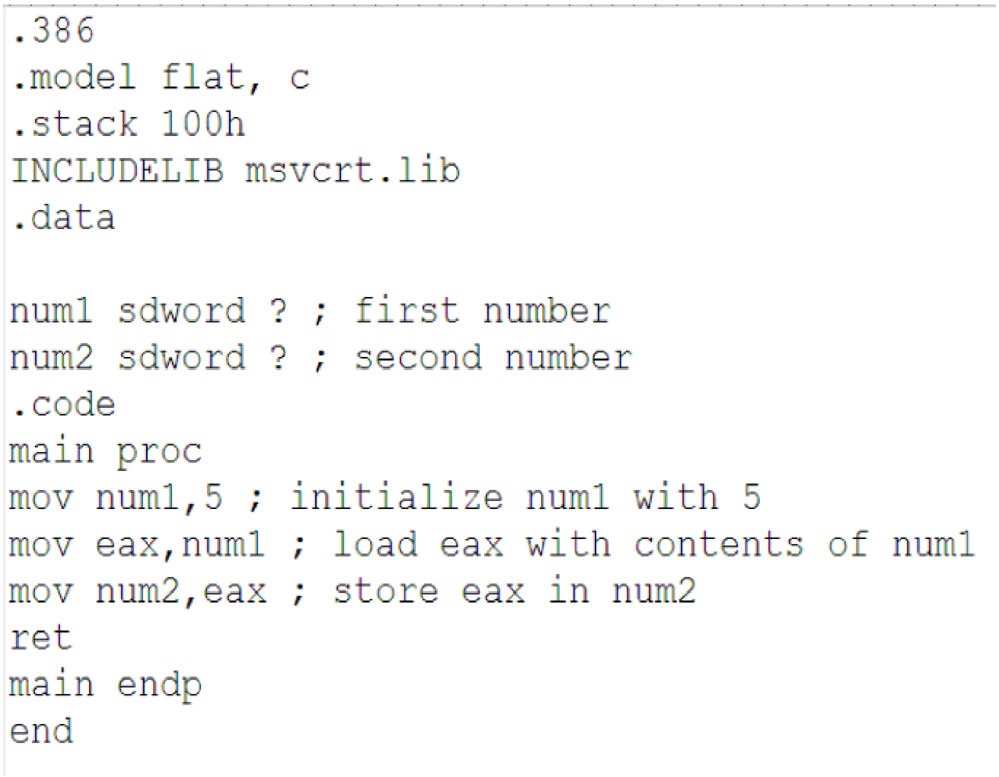
I. Introductory Lab

Conducted On

Friday, 13th September 2013

Lab Activity Instructions

1. Set up your Visual Studio .NET environment to assembling your code
2. Type in the following program
3. Remove any syntactic errors and make sure that it runs



```
.386
.model flat, c
.stack 100h
INCLUDELIB msvcrt.lib
.data

num1 sdword ? ; first number
num2 sdword ? ; second number
.code
main proc
mov num1,5 ; initialize num1 with 5
mov eax,num1 ; load eax with contents of num1
mov num2,eax ; store eax in num2
ret
main endp
end
```

Sample Solution

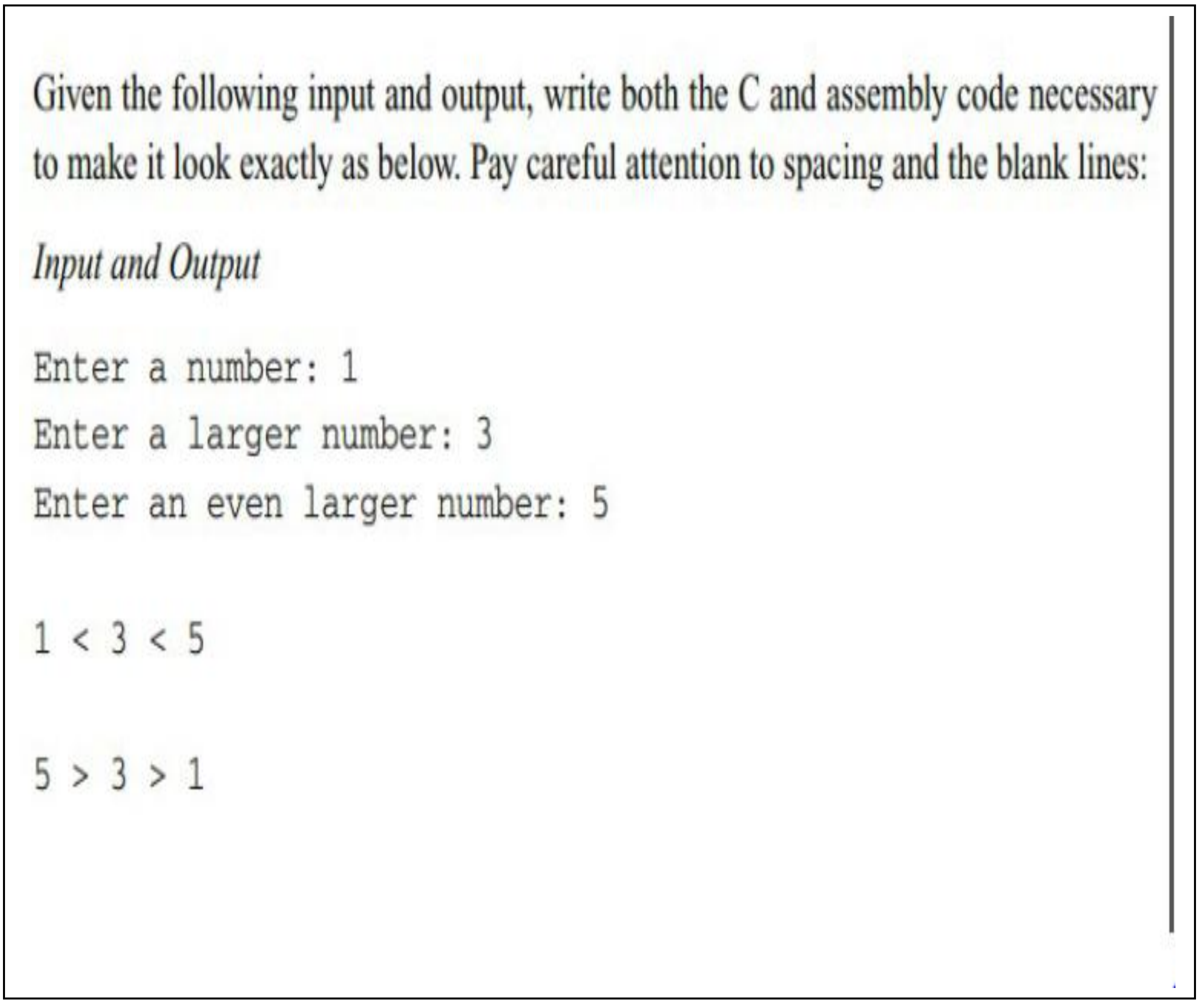
No solution, code just need to be typed in to enable students to understand h environment works.

II. Input / Output

Conducted On

Friday, 20th September 2013

Lab Activity Instructions



Given the following input and output, write both the C and assembly code necessary to make it look exactly as below. Pay careful attention to spacing and the blank lines:

Input and Output

```
Enter a number: 1
Enter a larger number: 3
Enter an even larger number: 5


1 < 3 < 5


5 > 3 > 1
```

Sample Solution

```
.386
.model flat, c
.stack 100h

INCLUDELIB msvcrt.lib

printf PROTO arg1:Ptr Byte, printlist: VARARG
scanf PROTO arg1:Ptr Byte, scanlist: VARARG

.data

p1fmt byte 0Ah, "%s", 0
p1msg byte "Enter the first number: ", 0

p2fmt byte "%s", 0
p2msg byte "Enter the second number: ", 0

s12fmt byte "%d", 0

p34fmt byte 0Ah, "%s%d", 0
p3msg byte "The sum of the two numbers is ", 0
p4msg byte "The difference of the two numbers is ", 0

fpfmt byte 0Ah, 0Ah, "%s", 0Ah, 0
fpmsg byte "Strike any character key to exit.", 0
fsfmt byte "%s", 0
dummy byte ?

num1 sdword 0
num2 sdword 0
sum sdword 0
diff sdword 0

.code

main proc

; get first number
INVOKE printf, ADDR p1fmt, ADDR p1msg
INVOKE scanf, ADDR s12fmt, ADDR num1
```

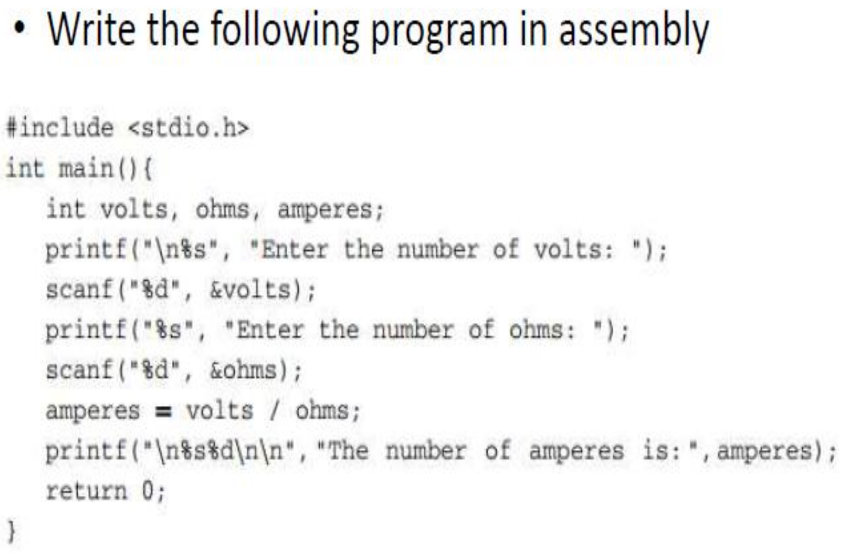
```
; get second number
INVOKE printf, ADDR p2fmt, ADDR p2msg
INVOKE scanf, ADDR s12fmt, ADDR num2

; add the numbers
mov eax, num1
add eax, num2
mov sum, eax

; subtract the numbers
mov eax, num1
sub eax, num2
mov diff, eax
; print the sum
INVOKE printf, ADDR p34fmt, ADDR p3msg, sum
; print the diff
INVOKE printf, ADDR p34fmt, ADDR p4msg, diff
; make an exit
INVOKE printf, ADDR fpfmt, ADDR fpmsg
INVOKE scanf, ADDR fsfmt, ADDR dummy
ret
main endp
```

III. Arithmetic Instructions – Addition & Subtraction

Conducted On

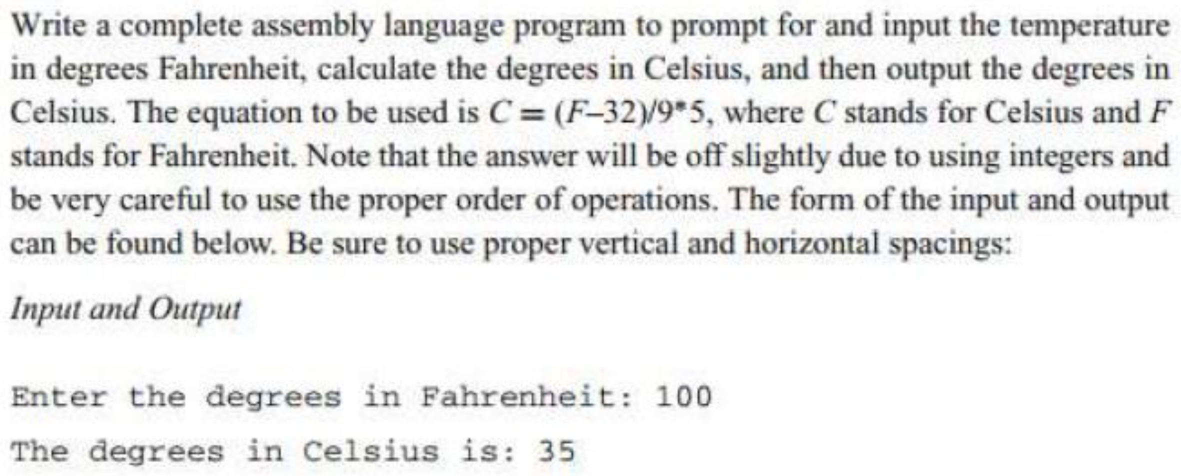
27th September 2013

Lab Activity Instructions



- Write the following program in assembly

```
#include <stdio.h>
int main(){
   int volts, ohms, amperes;
   printf("\n%s", "Enter the number of volts: ");
   scanf("%d", &volts);
   printf("%s", "Enter the number of ohms: ");
   scanf("%d", &ohms);
   amperes = volts / ohms;
   printf("\n%s%d\n\n","The number of amperes is:",amperes);
   return 0;
}
```



Write a complete assembly language program to prompt for and input the temperature in degrees Fahrenheit, calculate the degrees in Celsius, and then output the degrees in Celsius. The equation to be used is $C = (F-32)/9*5$, where C stands for Celsius and F stands for Fahrenheit. Note that the answer will be off slightly due to using integers and be very careful to use the proper order of operations. The form of the input and output can be found below. Be sure to use proper vertical and horizontal spacings:

Input and Output

```
Enter the degrees in Fahrenheit: 100
The degrees in Celsius is: 35
```

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib

printf PROTO arg1:ptr Byte,Printlist:VARARG
scanf PROTO arg2:ptr Byte,inputlist:VARARG

.data
            vol sdword ?
            ohm sdword ?
            amp sdword ?

            msg0fmt byte 0Ah,"%s",0
            in1fmt byte "%d",0
            msg2fmt byte 0Ah,"%s%d",0Ah,0Ah,0

            msg0 byte "Enter the Number of volts: ",0
            msg1 byte "Enter the Number of ohms: ",0
            msg2 byte "The number of amperes is: ",0
.code
main proc
            INVOKE printf,ADDR msg0fmt,ADDR msg0
            INVOKE scanf,ADDR in1fmt,ADDR vol
            INVOKE printf,ADDR msg0fmt,ADDR msg1
            INVOKE scanf,ADDR in1fmt,ADDR ohm

            mov eax,vol
            cdq
```

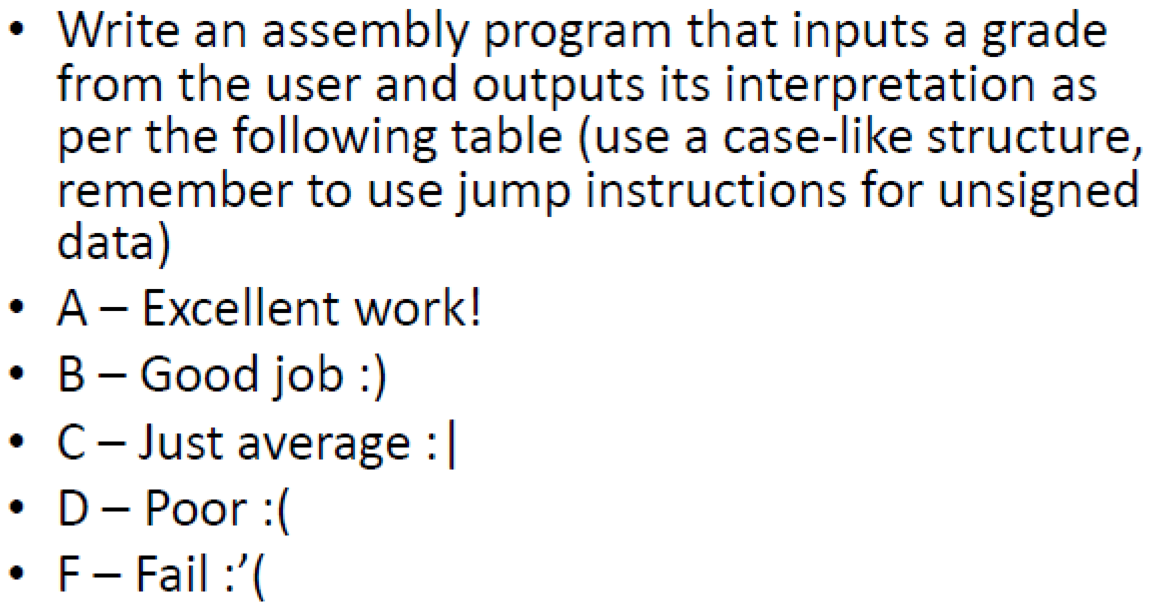
```
idiv ohm
mov amp,eax

INVOKE printf,ADDR msg2fmt,ADDR msg2,amp
ret
main endp
end

.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib

printf PROTO arg1:ptr Byte,Printlist:VARARG
scanf PROTO arg2:ptr Byte,inputlist:VARARG

.data
    cel sdword ?
    fah sdword ?
    nine sdword 9
    five sdword 5

        msg0fmt byte 0Ah,"%s",0
        in1fmt byte "%d",0
        msg1fmt byte 0Ah,"%s%d",0Ah,0

        msg0 byte "Enter the degrees of Fahrenheit: ",0
        msg1 byte "The degrees in Celsius: ",0
.code
main proc
        INVOKE printf,ADDR msg0fmt,ADDR msg0
        INVOKE scanf,ADDR in1fmt,ADDR fah

        mov ebx,fah
        sub ebx,32
        mov eax,ebx
        cdq
        idiv nine
        imul five
        mov cel,eax
        INVOKE printf,ADDR msg1fmt,ADDR msg1,cel
ret
main endp
end
```

IV. Selection Structures

Conducted On

Friday, 25th October 2013

Lab Activity Instructions



- Write an assembly program that inputs a grade from the user and outputs its interpretation as per the following table (use a case-like structure, remember to use jump instructions for unsigned data)
- A – Excellent work!
- B – Good job :)
- C – Just average :|
- D – Poor :(
- F – Fail :'(

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib
printf proto arg1:ptr byte, printlist :VARARG
scanf proto arg1:ptr byte, inputlist :VARARG
.data
msg0fmt byte "%s",0Ah,0
msg0 byte "Enter Grade: ",0
inputfmt byte "%s",0
msg1fmt byte "%s", 0Ah,0
msg1 byte "Excellent ",0
msg2fmt byte "%s", 0Ah,0
msg2 byte "Good ",0
msg3fmt byte "%s", 0Ah,0
msg3 byte "Average ",0
```

```
msg4fmt byte "%s", 0Ah,0
msg4 byte "Poor ",0
msg5fmt byte "%s", 0Ah,0
msg5 byte "Fail",0

grade byte ?

.code
main proc
INVOKE printf, ADDR msg0fmt, ADDR msg0
INVOKE scanf, ADDR inputfmt, ADDR grade
cmp grade,'A'
je else1
cmp grade,'B'
je else2
cmp grade,'C'
je else3
cmp grade, 'D'
je else4
cmp grade,'F'
je else5
else1:INVOKE printf,ADDR msg1fmt,ADDR msg1
jmp exit1
else2:INVOKE printf,ADDR msg2fmt,ADDR msg2
jmp exit2
else3:INVOKE printf,ADDR msg3fmt,ADDR msg3
jmp exit3
else4:INVOKE printf,ADDR msg4fmt,ADDR msg4
jmp exit4
else5:INVOKE printf,ADDR msg5fmt,ADDR msg5
jmp exit5
exit1:nop
exit2:nop
exit3:nop
exit4:nop
exit5:nop
main endp
end
```

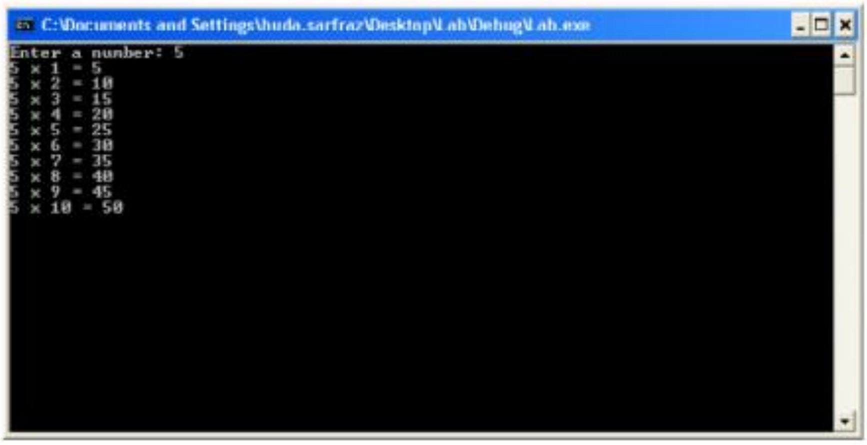
V. Iteration Structures

Conducted On

1st November 2013

Lab Activity Instructions

- Write an assembly program that takes a number as input from a user and then prints its multiplication table from 1 to 10


C:\Documents and Settings\huda.sarfraz\Desktop\Lab\Debug\Lab.exe

```
Enter a number: 5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

Sample Solution

```
.386
.model flat, c
.stack 100h

INCLUDELIB msvcrt.lib

printf PROTO arg1: Ptr Byte, printlist: VARARG
scanf PROTO arg1: Ptr Byte, scanlist: VARARG

.data

msg1fmt byte "%s", 0
msg1 byte "Enter a number: ", 0
msgmul byte " x ", 0
msgeq byte " = ", 0
```

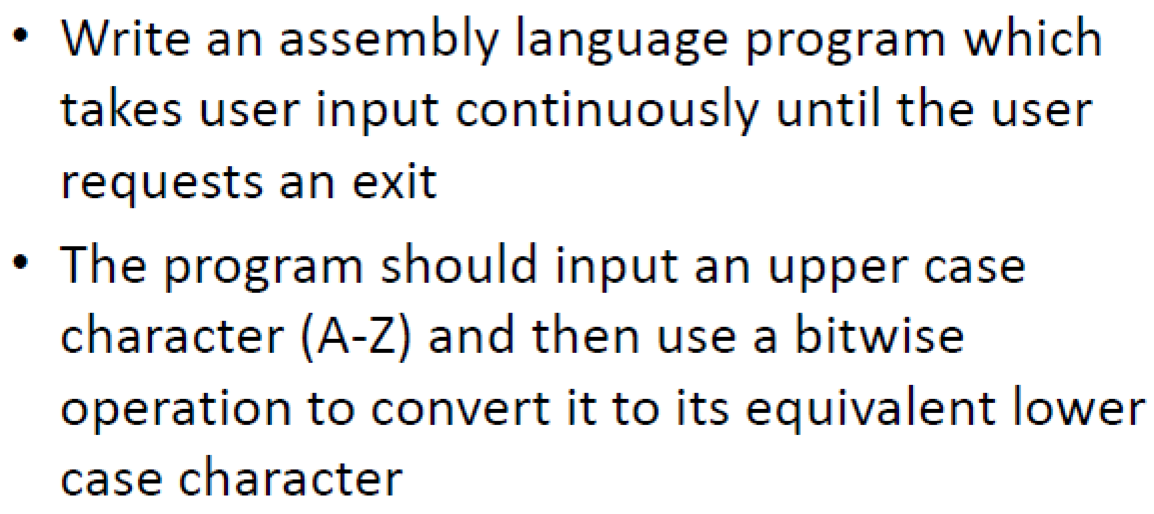
```
inputfmt byte "%d", 0


msgfmtnum byte "%d", 0
msgfmtnumn byte "%d", 0Ah, 0

usernum sdword ?
temp sdword ?
multiplier sdword 1


.code

main proc

INVOKE printf, ADDR msg1fmt, ADDR msg1
INVOKE scanf, ADDR inputfmt, ADDR usernum

mov ecx, 10
for01: nop
mov temp, ecx
INVOKE printf, ADDR msgfmtnum, usernum
INVOKE printf, ADDR msg1fmt, ADDR msgmul
INVOKE printf, ADDR msgfmtnum, multiplier
INVOKE printf, ADDR msg1fmt, ADDR msgeq
mov eax, multiplier
imul usernum
inc multiplier
INVOKE printf, ADDR msgfmtnumn, eax

mov ecx, temp
loop for01
endfor1: nop


ret
main endp
end
```

VI. Bitwise Operators

Conducted On

Tuesday, 5th November 2013

Lab Activity Instructions



- Write an assembly language program which takes user input continuously until the user requests an exit
- The program should input an upper case character (A-Z) and then use a bitwise operation to convert it to its equivalent lower case character

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib
printf PROTO arg1:ptr byte ,printlist :VARARG
scanf PROTO arg1:ptr byte ,inputlist :VARARG

.data
msg1fmt byte "%s",0Ah,0Ah,0
msg1 byte "Enter A Upper Case Letter: ",0
input byte "%s",0
msg2fmt byte "%s%s",0
msg2 byte "Lower Case Letter Is: ",0
msg3fmt byte"%s",0
msg3 byte "Input Value is Invalid ",0
 letter byte ?
 .code
main proc
```

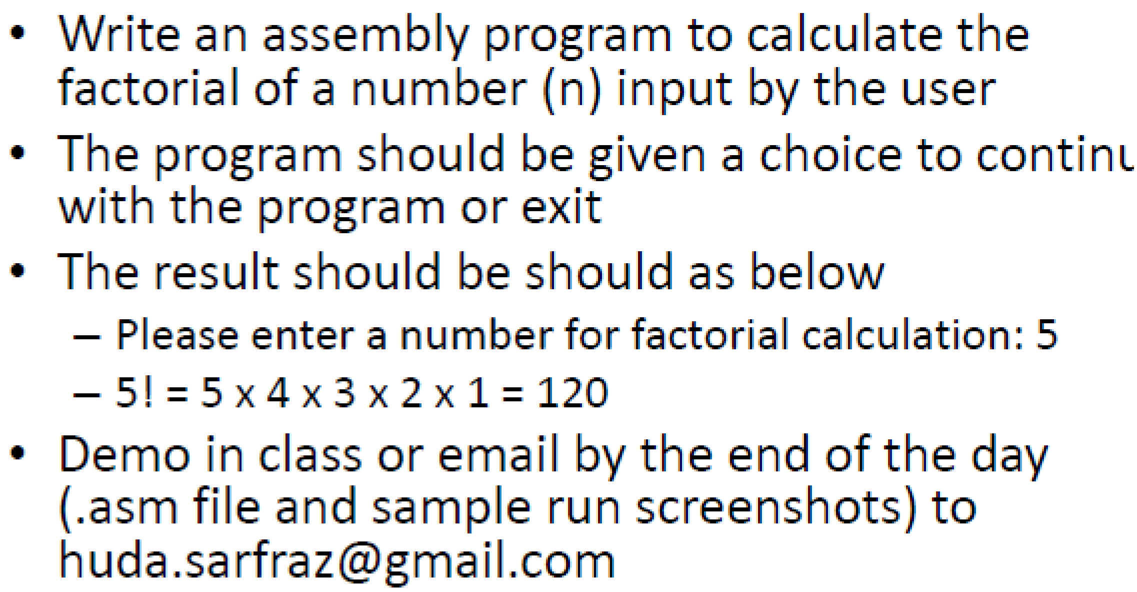
```
Invoke printf, ADDR msg1fmt, ADDR msg1
Invoke scanf, ADDR input, ADDR letter
        cmp letter,'A'
        ja else1
        cmp letter,'Z'
        jb end1
else1:or letter, 00100000b
        Invoke printf, ADDR msg2fmt,ADDR msg2,ADDR letter
        jmp end2
end1: nop
        Invoke printf, ADDR msg3fmt,ADDR msg3
        jmp end2
end2:nop
ret
main endp
end
```

VII. Debugging Practice

Conducted On

Friday, 8th November 2013

Lab Activity Instructions



- Write an assembly program to calculate the factorial of a number (n) input by the user
- The program should be given a choice to continu with the program or exit
- The result should be should as below
  - Please enter a number for factorial calculation: 5
  - 5! = 5 x 4 x 3 x 2 x 1 = 120
- Demo in class or email by the end of the day (.asm file and sample run screenshots) to huda.sarfraz@gmail.com

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib
printf PROTO arg1:ptr Byte,Printlist:VARARG
scanf PROTO arg2:ptr Byte,inputlist:VARARG
.data
                    msg0fmt byte 0Ah,"%s",0
                    msg0 byte "Enter a Number(-1 to Exit): ",0
                    num sdword ?
                    in0fmt byte "%d",0
                    msg1fmt byte "%d%s%d",0Ah,0
                    msg2fmt byte 0Ah,"%d%s",0
                    msg3fmt byte "%d",0
                    msg4fmt byte "%s",0
                    msg1 byte "! = ",0
                    msg2 byte " x ",0
                    msg3 byte " = ",0
```

```
                        result sdword 1



.code
main proc
while01:  INVOKE printf, ADDR msg0fmt,ADDR msg0
                           INVOKE scanf, ADDR in0fmt,ADDR num
                          mov result,1
if01:                   cmp num,-1
                          je end01
                          INVOKE printf, ADDR msg2fmt,num,ADDR msg1
else01:
       mov eax,num
                          cmp eax,1
                          jnle while02
                          INVOKE printf, ADDR msg1fmt,num,ADDR msg3,result
                          jmp while01

while02:  mov eax,num
                          cmp eax,1
                          je else01

                          INVOKE printf, ADDR msg3fmt,num
                          INVOKE printf, ADDR msg4fmt,ADDR msg2
                          mov eax,num
                          imul result
                          mov result,eax
                          dec num

                          jmp while02
end01:    nop

ret
main endp
end
```

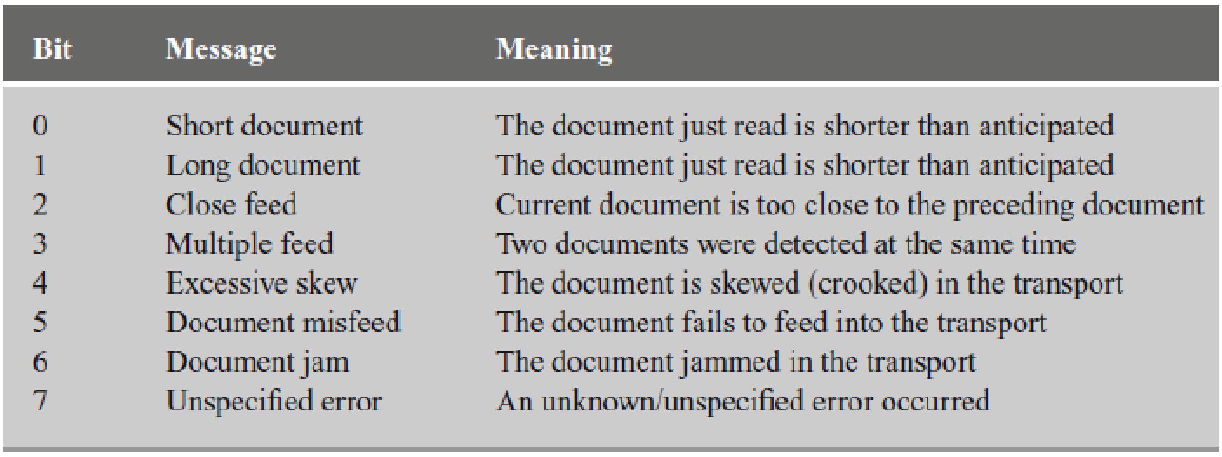
VIII. Bitwise & Stack Operations (OCR machine simulation)

Conducted On

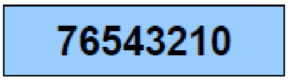
Tuesday, 19th November 2013

Lab Activity Instructions

1. Computers use individual bits in a memory or register to indicate the status or to control various parts of the CPU or peripheral devices
2. For example, an optical character recognition (OCR) device that reads typed or handwritten characters from paper
3. On larger scales, these machines may be handling more than one paper at a time (like a photocopy machine or a printer)
4. A memory location may be used to indicate the status of the paper in process
5. This is the document status byte (DSB)
6. In the simulation, we will declare a variable dsb to model this behavior: dsb dword ?

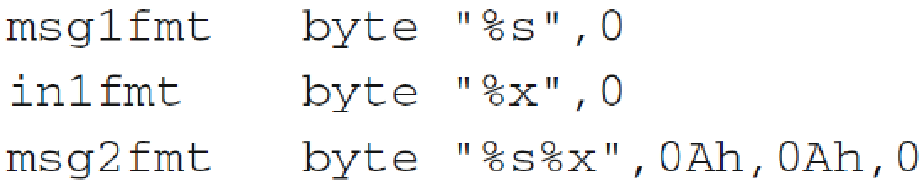


Bit	Message	Meaning
0	Short document	The document just read is shorter than anticipated
1	Long document	The document just read is shorter than anticipated
2	Close feed	Current document is too close to the preceding document
3	Multiple feed	Two documents were detected at the same time
4	Excessive skew	The document is skewed (crooked) in the transport
5	Document misfeed	The document fails to feed into the transport
6	Document jam	The document jammed in the transport
7	Unspecified error	An unknown/unspecified error occurred



76543210

7. To simulate input, we can input the dsb using hexadecimal format instead of binary



```
msg1fmt   byte "%s",0
in1fmt    byte "%x",0
msg2fmt   byte "%s%x",0Ah,0Ah,0
```

8. Use a switch case structure
9. For each case
    a. Push your value onto the stack
    b. Perform an AND operation with the appropriate bit pattern (AND dsb, bit pattern)
        i. 00000001b for short message
        ii. 00000010b for long message and so on
10. If you get something other than a zero, output the corresponding message for that state
11. Your program should keep asking for input in a loop

```
.386
.model flat,c
.stack 100h

INCLUDELIB msvcrt.lib

printf PROTO arg1:Ptr Byte,Printlist:VARARG
scanf PROTO arg2:Ptr Byte,Scanlist:VARARG

.data

dsb   dword ?
l    sdword ?


msg1fmt byte"%s",0
msg2fmt byte"%x",0
msg3fmt byte"%s%x",0
msg11   byte"Enter Hexadecimal: ",0Ah,0
msg10   byte"Your Hexadecimal is: ",0Ah,0
msg0    byte "Short Code",0Ah,0
msg1    byte "Long Document",0Ah,0
msg2    byte "Close Feed",0Ah,0
msg3    byte "Multiple Feed",0Ah,0
msg4    byte "Excessive Scew",0Ah,0
msg5    byte "Document Misfeed",0Ah,0
msg6    byte "Document Jam",0Ah,0
msg7    byte "Unspecified Error",0Ah,0

.code

main proc
```

```
INVOKE printf, ADDR msg1fmt, ADDR msg11
INVOKE scanf, ADDR msg2fmt,  ADDR dsb
INVOKE printf, ADDR msg1fmt, ADDR msg10, ADDR dsb

while1:
cmp dsb, 11111111b
jg end01



case0:
push dsb
and dsb, 00000001b
pop dsb
jz case1
INVOKE printf, ADDR msg1fmt, ADDR msg0

case1:
push dsb
and dsb, 00000010b
pop dsb
jz case2
INVOKE printf, ADDR msg1fmt, ADDR msg1

case2:
push dsb
and dsb, 00000100b
pop dsb
jz case3
INVOKE printf, ADDR msg1fmt, ADDR msg2

case3:
push dsb
and dsb, 00001000b
pop dsb
jz case4
INVOKE printf, ADDR msg1fmt, ADDR msg3

case4:
push dsb
and dsb, 00010000b
pop dsb
jz case5
INVOKE printf, ADDR msg1fmt, ADDR msg4
```

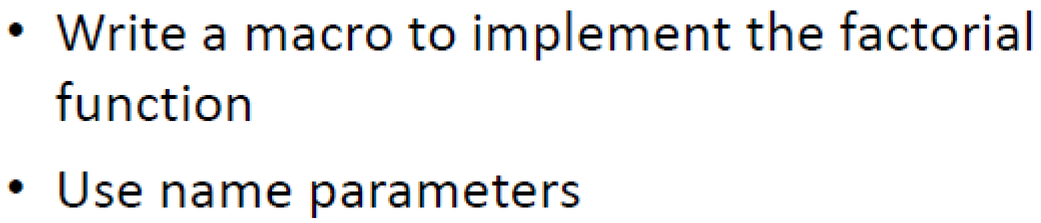
```
case5:
push dsb
and dsb, 00100000b
pop dsb
jz case6
INVOKE printf, ADDR msg1fmt, ADDR msg5

case6:
push dsb
and dsb, 01000000b
pop dsb
jz case7
INVOKE printf, ADDR msg1fmt, ADDR msg6


case7:
push dsb
and dsb, 10000000b
pop dsb
jz end02
INVOKE printf, ADDR msg1fmt, ADDR msg7
end02:
INVOKE printf, ADDR msg1fmt, ADDR msg11
INVOKE scanf, ADDR msg2fmt,  ADDR dsb
INVOKE printf, ADDR msg1fmt, ADDR msg10, ADDR dsb

jmp while1


end01: nop

ret
main endp
end
```

IX. Macros

Conducted On

26th November 2013

Lab Activity Instructions



- Write a macro to implement the factorial function
- Use name parameters

Sample Solution

```
     .386
           .model flat ,c
           .stack 100h
           INCLUDELIB msvcrt.LIB
printf       PROTO arg1:ptr byte, printlist:VARARG
scanf        PROTO arg2:ptr byte, scanlist:VARARG
           .data

                 msg1fmt      byte 0AH,"%s",0AH,0
           msg2fmt      byte "%d",0
                 msg3        byte 0AH,"Enter y:  ",0
                 msg4        byte 0AH,"Enter z:  ",0
                 msg5        byte 0AH,"result is :  ",0
                 y sdword ?
                 z sdword ?
                 m sdword ?
                 d sdword ?
                 val sdword ?
                 s sdword ?
                 .code
                 main        proc
           INVOKE printf, ADDR msg1fmt , ADDR msg3
                 INVOKE scanf, ADDR msg2fmt, ADDR y
```

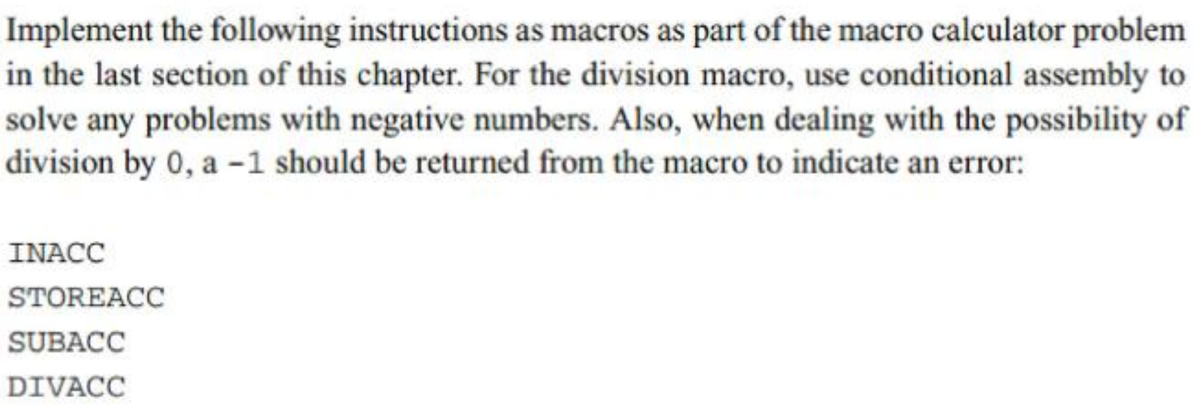
```
                                        INVOKE printf, ADDR msg1fmt , ADDR msg4
                                        INVOKE scanf, ADDR msg2fmt, ADDR z

                                        mov eax,y
                                        sal eax ,3
                                        mov m,eax

                                        mov eax,z


                                        sar eax,1
                                        add eax,m

                                        mov val,eax




                                        INVOKE printf ,ADDR msg1fmt, ADDR msg5
                                        INVOKE printf, ADDR msg2fmt, val

                                        INVOKE scanf, ADDR msg2fmt, ADDR s
                                        ret
                  main        endp
                        end
```

X. Macros

Conducted On

Friday 5th December 2013

Lab Activity Instructions



Implement the following instructions as macros as part of the macro calculator problem in the last section of this chapter. For the division macro, use conditional assembly to solve any problems with negative numbers. Also, when dealing with the possibility of division by 0, a -1 should be returned from the macro to indicate an error:

```
INACC
STOREACC
SUBACC
DIVACC
```

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib

printf PROTO arg1:ptr Byte, printlist:VARARG
scanf PROTO arg2:ptr Byte, inputlist:VARARG

.data
msg1fmt         byte 0Ah,"%s%d",0Ah,0Ah,0
msg1    byte "The contents of the accumulator are: ",0
msg2fmt         byte 0Ah,"%s",0
msg2    byte "Enter the content of the accumulator: ",0
in1fmt  byte "%d",0
temp    sdword ?
three   sdword 3
store   sdword ?
.code
LOADACC     macro operand
            mov eax,operand
```

```
endm
ADDACC    macro operand
          add eax,operand
endm
SUBACC    macro operand
          sub eax,operand
endm
STOREACC  macro
          mov store,eax
endm
MULTACC   macro operand
          push ebx
          push ecx
          mov ebx,eax
          mov eax,0
          mov ecx,operand
          if operand LT 0
          neg ecx
          endif
          .while ecx>0
          add eax,ebx
          dec ecx
          .endw
          if operand LT 0
          neg eax
          endif
          pop ecx
          pop ebx
endm
DIVACC    macro operand
          push ebx
          push ecx
          mov ebx,eax
          mov eax,operand
          mov ecx,0
    .if ebx>eax
          mov eax,0
          .endif
          .while eax>0
          sub eax,ebx
          inc ecx
          .endw
    mov eax,ecx
          pop ecx
```

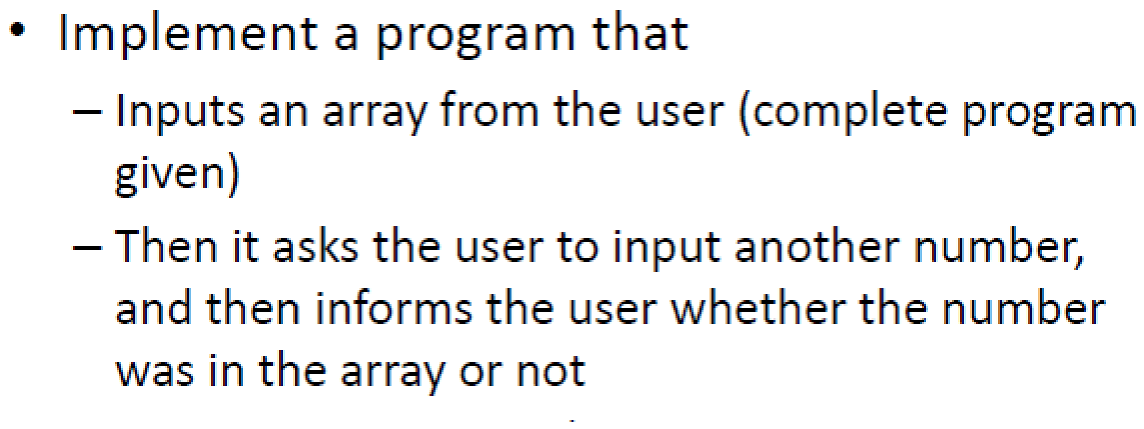
```
                        pop ebx
endm

main proc
                        LOADACC 8
                        ADDACC 2
                        SUBACC 3
                        CALL OUTACC
                        CALL INACC
                        ADDACC three
                        MULTACC 4
                        DIVACC 128
                        CALL OUTACC

ret
main endp
OUTACC proc
                        push eax
                        push ecx
                        push edx
                        mov temp,eax
                        INVOKE printf,ADDR msg1fmt,ADDR msg1,temp
                        pop edx
                        pop ecx
                        pop eax
                        ret
OUTACC  endp
INACC proc
                        push ebx
                        push ecx
                        push edx
                        INVOKE printf,ADDR msg2fmt,ADDR msg2
                        INVOKE scanf,ADDR in1fmt,ADDR temp
                        mov eax,temp
                        pop edx
                        pop ecx
                        pop ebx
                        ret
INACC  endp
end
```

XI. Arrays

Conducted On

Friday 13th December 2013

Lab Activity Instructions



- Implement a program that
  - Inputs an array from the user (complete program given)
  - Then it asks the user to input another number, and then informs the user whether the number was in the array or not

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib

printf PROTO arg1:ptr Byte, printlist:VARARG
scanf PROTO arg2:ptr Byte, inputlist:VARARG

.data
        n sdword ?
        numarry sdword 20 dup(?)
        msg1fmt byte 0Ah,"%s",0
        in1fmt byte "%d",0
        msg2fmt byte "%d",0Ah,0
        msg1 byte "Enter size of array: ",0
        msg2 byte "Enter digit: ",0
        msg3 byte "Enter a digit to find from array: ",0
        msg4 byte "Number is in array ",0Ah,0
        msg5 byte "Number is not in array ",0Ah,0
        F sdword ?
```

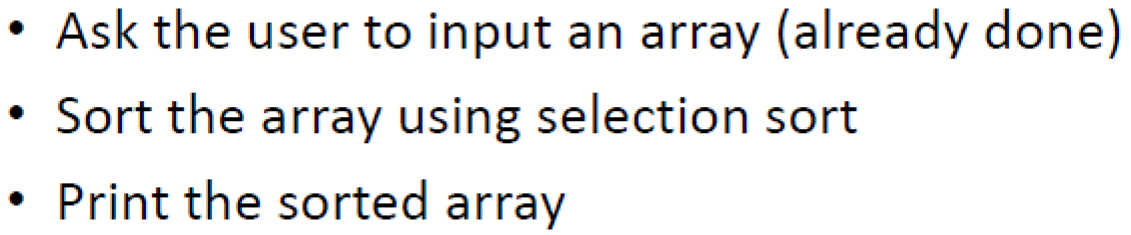
```
.code

main proc
        INVOKE printf,ADDR msg1fmt,ADDR msg1
        INVOKE scanf,ADDR in1fmt,ADDR n
        mov ecx,n
        mov ebx,0
        .if ecx>0
        .repeat
        push ecx
        push ebx
        INVOKE printf,ADDR msg1fmt,ADDR msg2
        INVOKE scanf,ADDR in1fmt,ADDR numarry[ebx]
        pop ebx
        pop ecx
        add ebx,4
        .untilcxz
        .endif

        INVOKE printf,ADDR msg1fmt,ADDR msg3
        INVOKE scanf,ADDR in1fmt,ADDR F
        mov ecx,n
        mov ebx,0
        mov edx,F
        .if ecx>0
        .repeat
        cmp edx,numarry[ebx]
        je iffound
        add ebx,4
        .untilcxz
        .endif
ifnotfound:INVOKE printf,ADDR msg1fmt,ADDR msg5
         jmp endif01
iffound:INVOKE printf,ADDR msg1fmt,ADDR msg4
                jmp endif01
endif01:nop
ret
main endp

end
```

XII. Selection Sort Using Arrays

Conducted On

Friday 20th December 2013

Lab Activity Instructions



- Ask the user to input an array (already done)
- Sort the array using selection sort
- Print the sorted array

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib

printf PROTO arg1:ptr Byte, printlist:VARARG
scanf PROTO arg2:ptr Byte, inputlist:VARARG

.data
        n sdword ?
        numarry sdword 20 dup(?)
        msg1fmt byte "%s",0
        in1fmt byte "%d",0
        msg1 byte "Enter size of Array: ",0
        msg2 byte "Enter Digit: ",0
        msg2fmt byte "%d    ",0
        Newline byte 0Ah,0
        msg3fmt byte 0Ah,"%s",0Ah,0
        msg4 byte "Before Selection Sort",0
        msg5 byte "After Selection Sort",0
.code
main proc
                call INPUTp
```

```
                    INVOKE printf,ADDR msg3fmt,ADDR msg4
                    call OUTPUTp
                    call SelectionSort
                    INVOKE printf,ADDR msg3fmt,ADDR msg5
                    call OUTPUTp
ret
main endp
INPUTp proc
                              INVOKE printf,ADDR msg1fmt,ADDR msg1
                              INVOKE scanf,ADDR in1fmt,ADDR n
                              mov ecx,n
                              mov ebx,0
                              .if ecx>0
                              .repeat
                              push ecx
                              push ebx
                              INVOKE printf,ADDR msg1fmt,ADDR msg2
                              INVOKE scanf,ADDR in1fmt,ADDR numarry[ebx]
                              pop ebx
                              pop ecx
                              add ebx,4
                              .untilcxz
                              ret
                              .endif
INPUTp endp

SelectionSort Proc
                              pushad
                              mov ecx,n
                              mov ebx,0
                              dec ecx
                              .repeat
                              push ecx
                              lea esi,numarry[ebx]
                              mov edi,esi
                              mov eax,[esi]

                                        .repeat
                                        push ecx
                                                  add edi,4
                                                  .if eax>[edi]
                                                            xchg
eax,[edi]
```

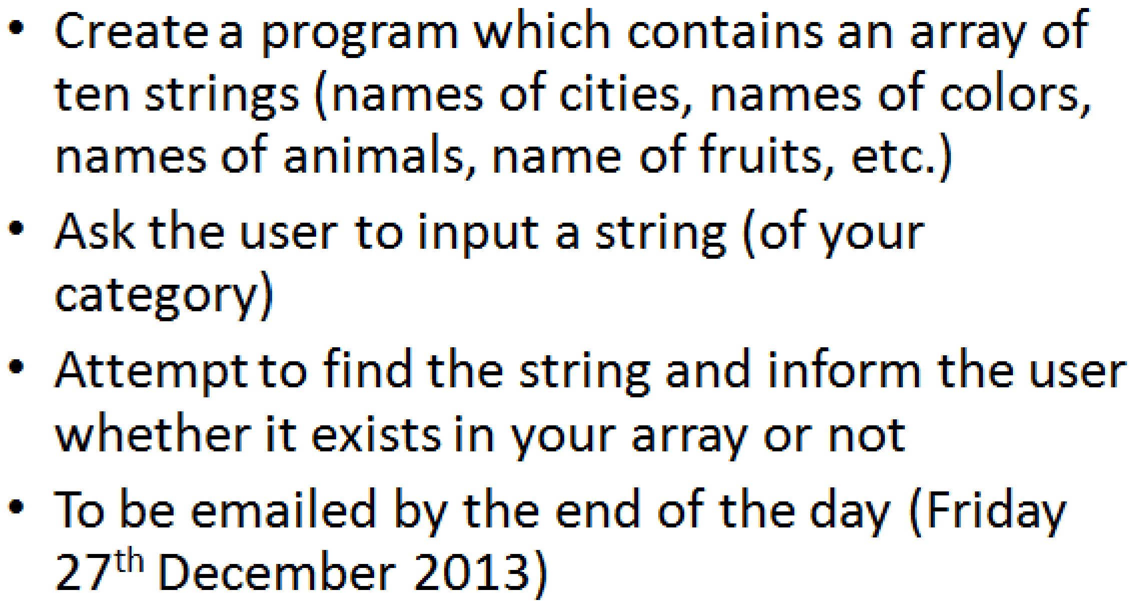
```
                                                                    mov
[esi],eax
   .endif

            pop ecx
           .untilcxz
                      add ebx,4
                       pop ecx
                      .untilcxz
                        popad
                          ret
SelectionSort endp

OUTPUTp proc
	mov ecx,n
	mov ebx,0
	.repeat
	push ecx
	push ebx
	INVOKE printf,ADDR msg2fmt,numarry[ebx]
	pop ebx
	pop ecx
	add ebx,4
	.untilcxz
	INVOKE printf,ADDR msg1fmt,ADDR Newline
	ret
OUTPUTp endp

end
```

XIII. 2D Arrays

Conducted On

Friday 27th December 2013

Lab Activity Instructions



- Create a program which contains an array of ten strings (names of cities, names of colors, names of animals, name of fruits, etc.)
- Ask the user to input a string (of your category)
- Attempt to find the string and inform the user whether it exists in your array or not
- To be emailed by the end of the day (Friday 27th December 2013)

Sample Solution

```
.386
.model flat,c
.stack 100h
INCLUDELIB msvcrt.lib

printf PROTO arg1:ptr Byte, printlist:VARARG
scanf PROTO arg2:ptr Byte, inputlist:VARARG

.data
                fruit byte "Apple","Grape","Guava","Mango","Peach"
                find byte 10 dup(?)
                msg1fmt byte "%s",0
                msg2fmt byte "%s",0Ah,0
                in1fmt byte "%s",0
                msg1 byte "Enter a fruit name: ",0
                msg2 byte "Found in String ",0Ah,0
                msg3 byte "Not found in String ",0Ah,0
                hold byte 10 dup(?)
                check sdword 1
.code
main proc
                call INPUT ;input what is found
                mov eax,0
                mov ecx,5
                .repeat
           push ecx
                push eax
                call getfruit ;Get fruit name from
                                ;check if same
                mov ecx,5
                lea esi,hold
                lea edi,find
                cld
                repe cmpsb
                .if ecx==0   ;check if found
                        mov check,ecx
                .endif
                pop eax
                add eax,5
                pop ecx
                .untilcxz
                .if check==0
                INVOKE printf,ADDR msg1fmt,ADDR msg2
                .else
                INVOKE printf,ADDR msg1fmt,ADDR msg3
                .endif

ret
main endp
INPUT proc
                INVOKE printf,ADDR msg1fmt,ADDR msg1
                INVOKE scanf,ADDR in1fmt,ADDR find
ret
INPUT endp
getfruit proc
        push ecx
                mov ecx,5
                lea esi,fruit
                add esi,eax
                lea edi,hold
                cld
                rep movsb
        pop ecx
ret
getfruit endp
end
```

Introduction to Programming

[LAB MANUAL]

Credit Hrs: 4 (Lec 2, Lab 1)
Semester: Spring 2015

Instructor: Sameen Reza

Lab 1: IDE basic Input and Output 3rdFeb, 2015

Type the following code, compile and run with different inputs shown below

Q 1:
```
#include <iostream>
using namespace std;
main()
{
        doublecurrentpay = 0.0;
        doubleraiserate = 0.0;
        double raise = 0.0;
        doublenewpay = 0.0;
        cout<<"enter current weekly pay:";
        cin>>currentpay;
        cout<<"enter raiserate" ;
        cin>>raiserate;
        raise = currentpay * raiserate;
        newpay = raise + currentpay;
        cout<<"newpay: " <<newpay<<endl;
}
```

Give the following inputs and write corresponding o utputs

Current weekly pay = 1000 **newpay:= __________________**
Rate raise = 0.25

Current weekly pay = 5000 **new pay := _________________**
Rate raise = 0.20

Current weekly pay = 1657.90 **new pay := __________________**
Rate raise = 0.10

Q2:
```
#include <iostream>
using namespace std;
main()
{
        int number1;
        int number2;

        cout<<"enter two numbers to comapre";
        cin>> number1 >> number2;

        if(number1==number2)
        cout<<number1<<"==" <<number2<<endl;
        if(number1!=number2)
        cout<<number1<< "!"<<number2<<endl;

        if(number1<number2)
        cout<<number1<<"<" <<number2<<endl;
}
```

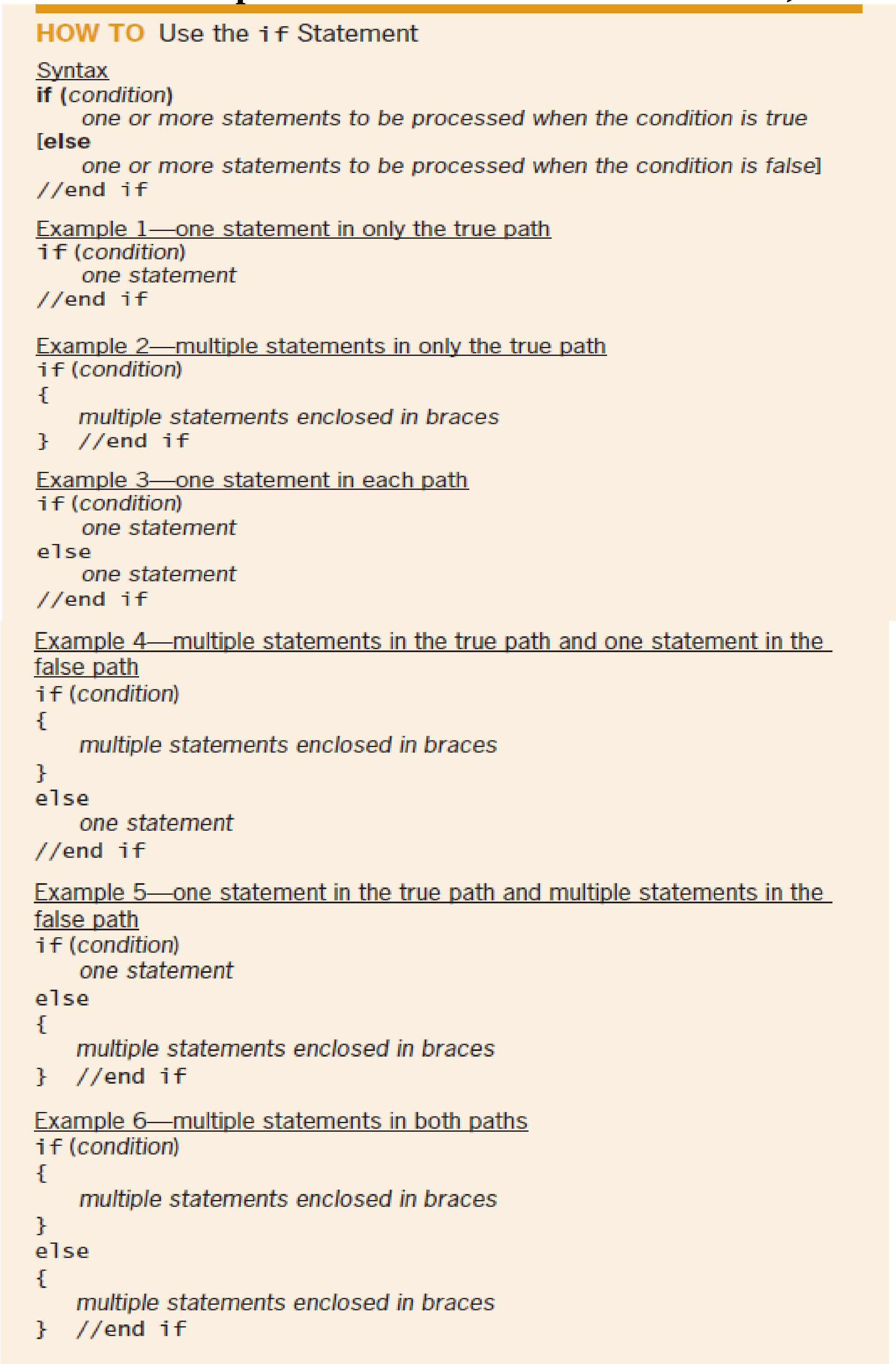
Give the following inputs and write corresponding outputs
Number 1 = _______ output:= _________________
Number 2 = _______

Number 1 = ______ output := ________________
Number 2 = _______

Number 1 = _______ output := _________________
Number 2 = _____________

Questions 1 , 2 ,3 ,4 from page 91- 94 of text book

Lab II: Simple If Else Statement 10th Feb, 2015



HOW TO Use the `if` Statement

Syntax

```
if (condition)
     one or more statements to be processed when the condition is true
[else
     one or more statements to be processed when the condition is false]
//end if
```

Example 1—one statement in only the true path

```
if (condition)
     one statement
//end if
```

Example 2—multiple statements in only the true path

```
if (condition)
{
     multiple statements enclosed in braces
}  //end if
```

Example 3—one statement in each path

```
if (condition)
     one statement
else
     one statement
//end if
```

Example 4—multiple statements in the true path and one statement in the false path

```
if (condition)
{
     multiple statements enclosed in braces
}
else
     one statement
//end if
```

Example 5—one statement in the true path and multiple statements in the false path

```
if (condition)
     one statement
else
{
     multiple statements enclosed in braces
}  //end if
```

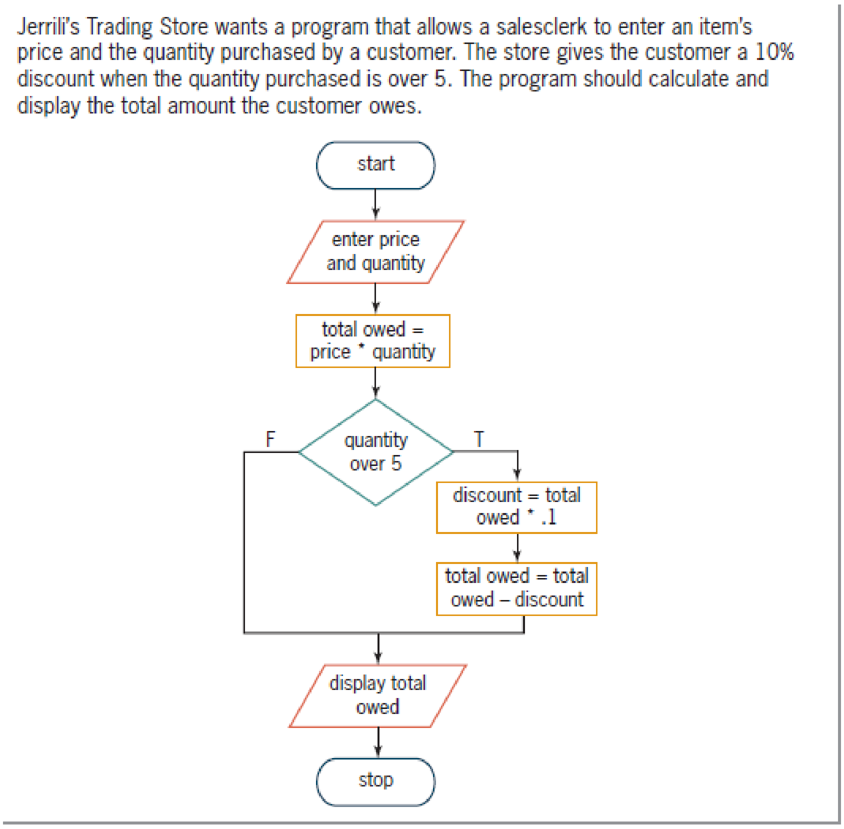
Example 6—multiple statements in both paths

```
if (condition)
{
     multiple statements enclosed in braces
}
else
{
     multiple statements enclosed in braces
}  //end if
```

Q1.Type the following code and check its output

```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
 {
        double salary = 0.0;
        double rate = 0.0;
        charpayGrade = ' ';

        cout<< "Current salary: ";
        cin>> salary;
        cout<< "Pay grade (1, 2, or 3): ";
        cin>>payGrade;

        if (payGrade == '1')
                rate = .03;
        else
                rate = .02;

        salary = salary + salary * rate;
        cout<< fixed <<setprecision(2);
        cout<< "New salary: " << salary <<endl;

        system("pause");
        return 0;
} //end of main function
```

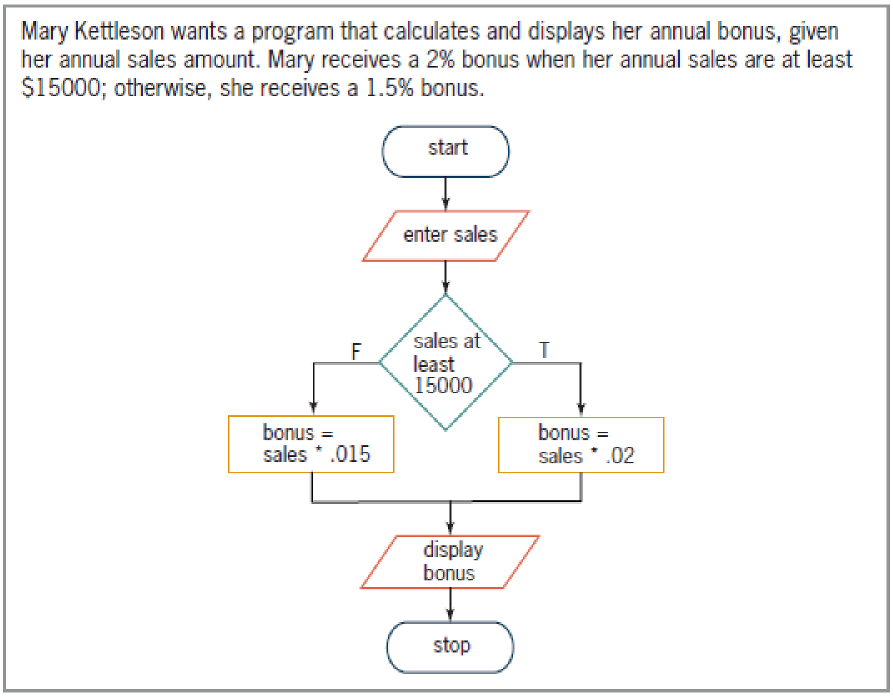
Q2. Write C++ code appropriate to the following flowchart. Check using specific inputs and outputs.



Jerrili's Trading Store wants a program that allows a salesclerk to enter an item's price and the quantity purchased by a customer. The store gives the customer a 10% discount when the quantity purchased is over 5. The program should calculate and display the total amount the customer owes.

start
enter price and quantity
total owed = price * quantity
quantity over 5
F
T
discount = total owed * .1
total owed = total owed – discount
display total owed
stop

Q3. Write C++ code appropriate to the following flowchart. Check using specific inputs and outputs.



Mary Kettleson wants a program that calculates and displays her annual bonus, given her annual sales amount. Mary receives a 2% bonus when her annual sales are at least $15000; otherwise, she receives a 1.5% bonus.

SUBMISSION SHEET

.1Q

Use Pay Grade “1”: __________salary:______________

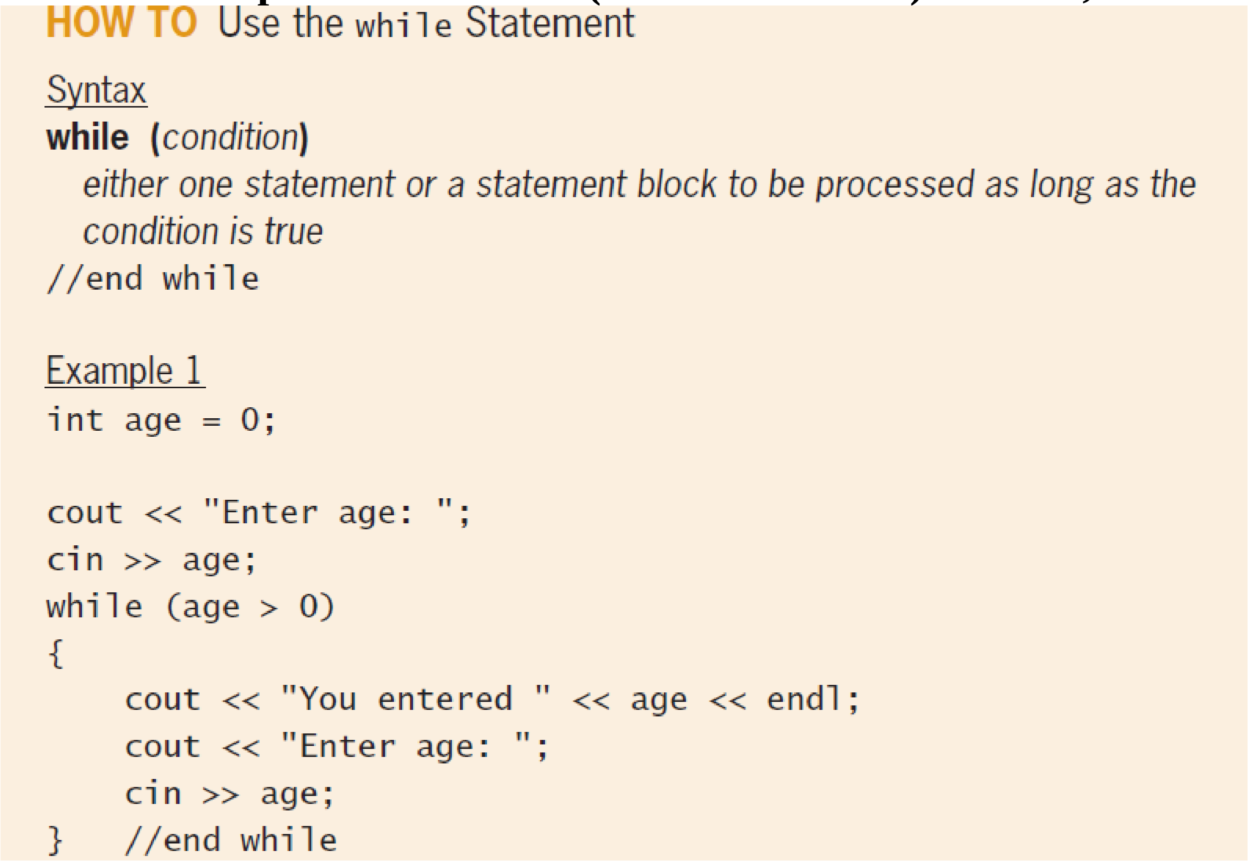
salary:______________ Use Pay Grade “3”:___________

Use Pay Grade: “5”:___________ salary:______________

```
#include <iostream>.2Q
#include <iomanip>
using namespace std;

int main()
{
//declarations
______________________________________________________________
______________________________________________________________
______________________________________________________________
__________________________________________________

//Take input
______________________________________________________________
______________________________________________________________
______________________________________________________________
__________________________________________________
//Calculate total owed

___________________________________________________________

//check quantity over 5?
______________________________________________________________
______________________________________________________________
______________________________________________________________
______________________________________________________________
______________________________________________________________
____________________________________________

//display total owed
____________________________________________________________________
____________________________________________________________
```

```
                                                  system("pause");
return 0;
} //end of main function
.3Q
#include <iostream>
#include <iomanip>
using namespace std;

int main()
 {
//declarations…used named constants in this program where required
____________________________________________________________

____________________________________________________________

____________________________________________________________

______________________________________________


//Take input
________________________________________________________

//check sales over 15000? (True and False functionality here)
____________________________________________________________

____________________________________________________________

____________________________________________________________

______________________________________________

____________________________________________________________

____________________________________________________________

____________________________________________________________

______________________________________________

//display bonus
____________________________________________________________

_____________________________________________________

system("pause");
return 0;
                                                  } //end of main function
```

Lab # 3: Repetition structure (while Statements) 17th Feb, 2015



HOW TO Use the `while` Statement

Syntax

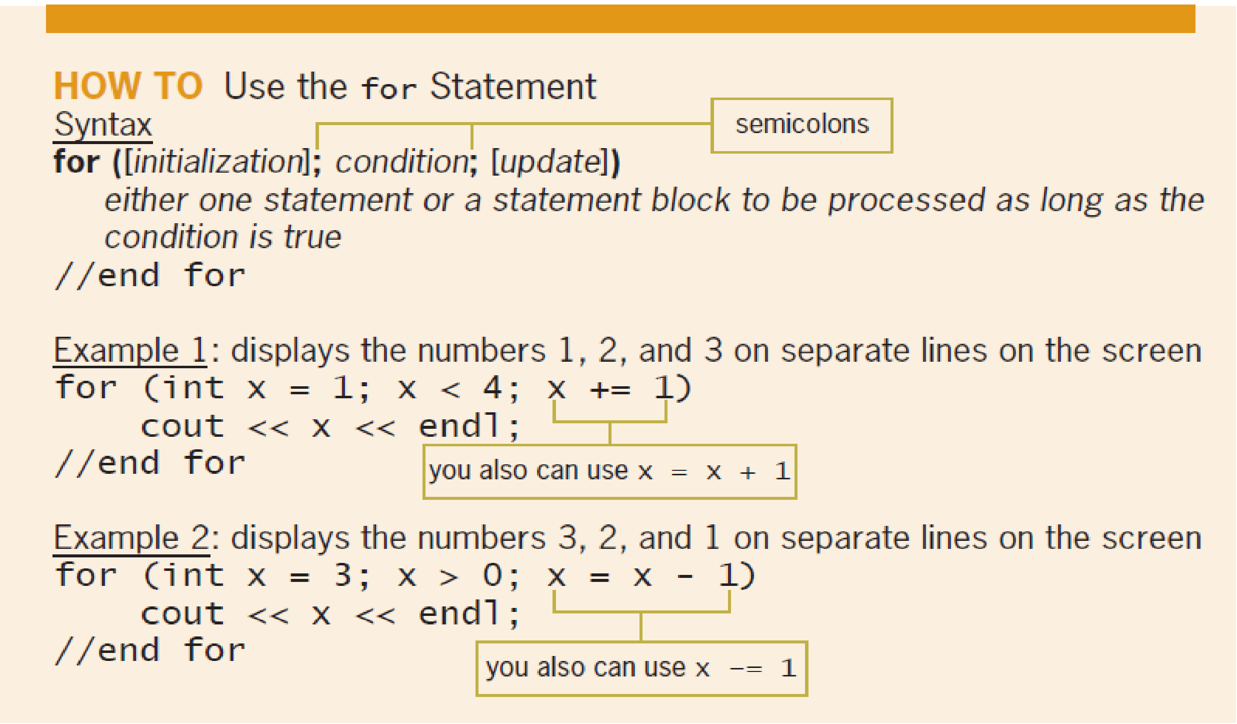
while (*condition*)

either one statement or a statement block to be processed as long as the condition is true

`//end while`

Example 1

```
int age = 0;

cout << "Enter age: ";
cin >> age;
while (age > 0)
{
    cout << "You entered " << age << endl;
    cout << "Enter age: ";
    cin >> age;
}   //end while
```



HOW TO Use the `for` Statement

Syntax

for ([*initialization*]; *condition*; [*update*])

either one statement or a statement block to be processed as long as the condition is true

`//end for`

semicolons

Example 1: displays the numbers 1, 2, and 3 on separate lines on the screen

```
for (int x = 1; x < 4; x += 1)
     cout << x << endl;
//end for
```

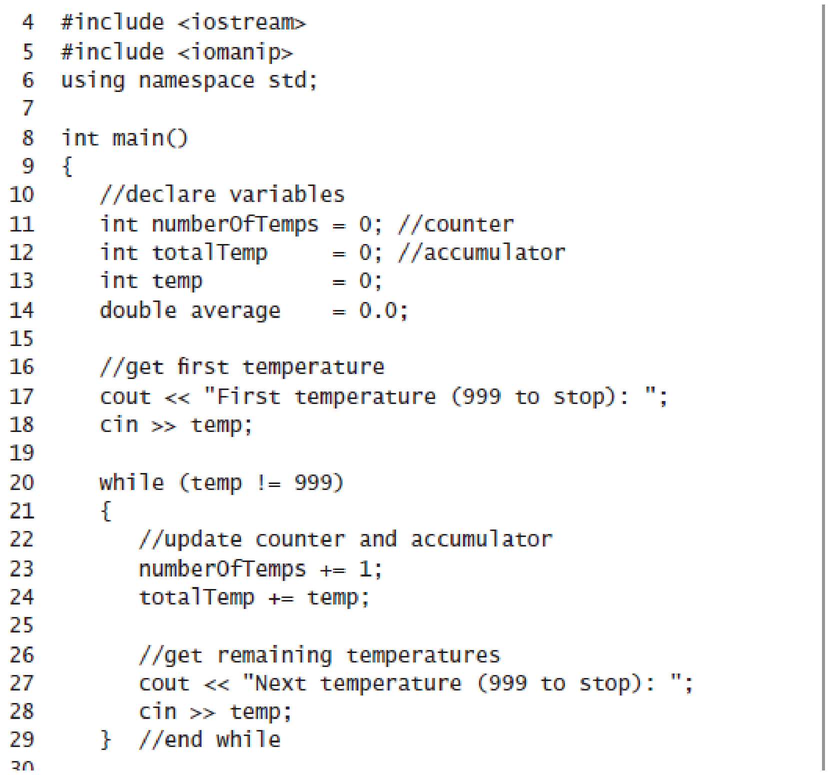
you also can use `x = x + 1`

Example 2: displays the numbers 3, 2, and 1 on separate lines on the screen

```
for (int x = 3; x > 0; x = x - 1)
     cout << x << endl;
//end for
```

you also can use `x -= 1`

Q1. Type the following C++ code (excluding the line numbers given on extreme left). Complete the code to add statements after the loop to calculate display the average. In case the loop has not executed even once the program should display appropriate message.



```
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
   //declare variables
   int numberOfTemps = 0; //counter
   int totalTemp     = 0; //accumulator
   int temp          = 0;
   double average    = 0.0;

   //get first temperature
   cout << "First temperature (999 to stop): ";
   cin >> temp;

   while (temp != 999)
   {
      //update counter and accumulator
      numberOfTemps += 1;
      totalTemp += temp;

      //get remaining temperatures
      cout << "Next temperature (999 to stop): ";
      cin >> temp;
   }  //end while
```

Q2. Change the while statement to a for statement. Save and then run the program. Enter the sentinel value. What does the program display?

Q3. Run the program again. Enter the following temperatures and sentinel value: -3, 32, -10, 40, and 999. What is the average temperature?

Q4. The sales manager at Jasper Music Company wants a program that allows him to enter the quarterly sales amount made in each of three regions: Region 1, Region 2, and Region 3. The program should calculate the total quarterly sales and then display the result on the screen. The program will use a counter to ensure that the sales manager enters exactly three sales amounts. It will use an accumulator to total the sales amounts.

Q5. Professor Chang wants a program that allows him to enter a student s project and test scores. The professor assigns 5 tests. Each test is worth 100 points. The program should calculate and display the total points the student earned on the projects and tests. It also should display the student’ s grade, using the scale below

Total points earned	**Grade**
350-500	A
315 – 349	B
280 – 314	C
245 – 279	D
210 – 244	E
200- 209	R
below 200	F

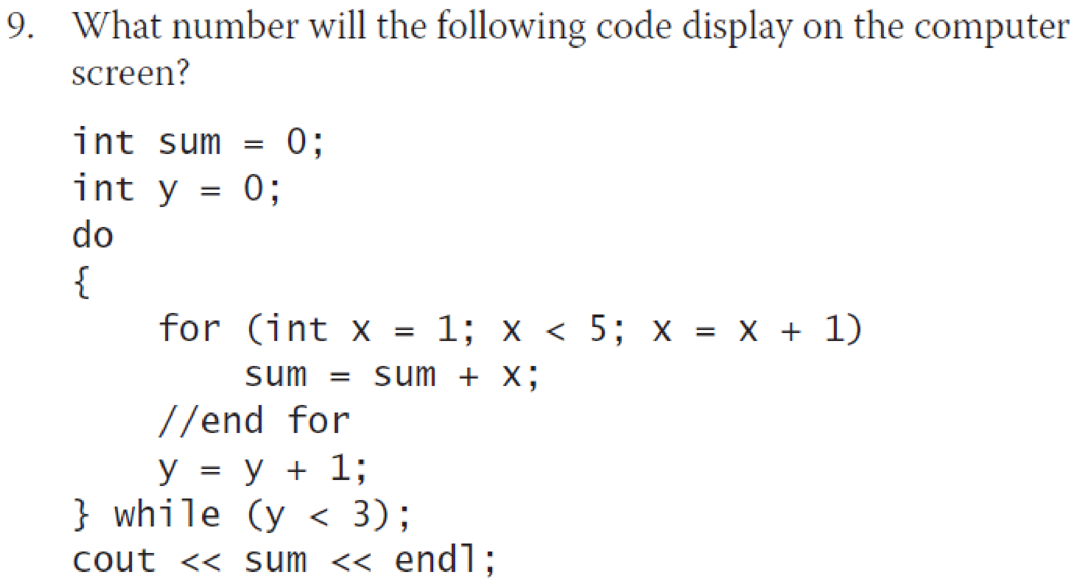
Example 1

Project and test scores: 45, 40, 41, 96, 89
Total points earned and grade: 311, B

Q6. Calculate factorial of a number. The number should be input from the user. Factorial of n = n(n-1)(n-2)…1. Factorial of 0 is 1 and Factorial of 1 is also 1.

Lab 4: Repetition Structures (nested loops and do-while loops) 24th feb, 2015

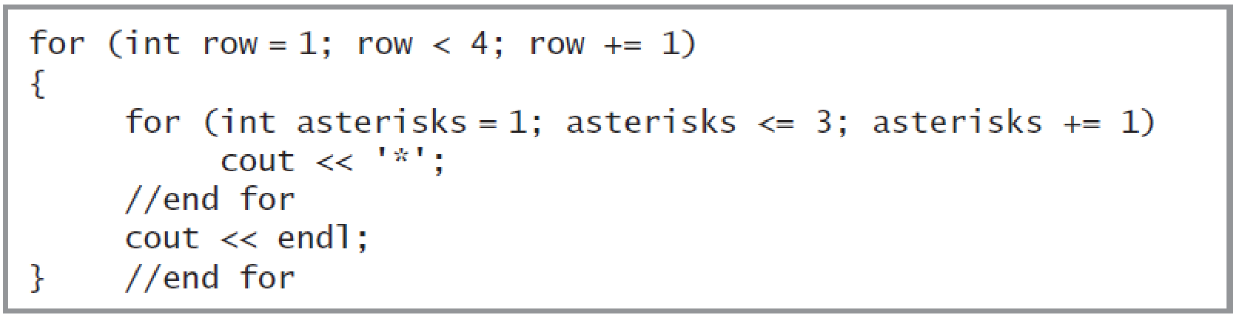
Q1.



9. What number will the following code display on the computer screen?

```
int sum = 0;
int y = 0;
do
{
    for (int x = 1; x < 5; x = x + 1)
        sum = sum + x;
    //end for
    y = y + 1;
} while (y < 3);
cout << sum << endl;
```

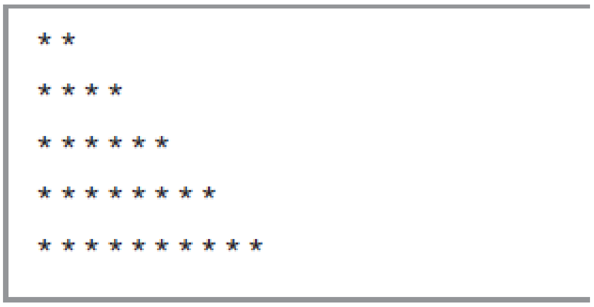
Q2. The code below prints a box of asterisks. Type the code and make changes to it so that it prints a triangle of stars such that the first row contains 1 start, 2nd row has 2 stars, and so on.



```
for (int row = 1; row < 4; row += 1)
{
    for (int asterisks = 1; asterisks <= 3; asterisks += 1)
        cout << '*';
    //end for
    cout << endl;
}   //end for
```

Q3. Write code to print the following pattern of stars.

a) Note that first line contain 2 stars, 2nd line contains 4 stars and so on



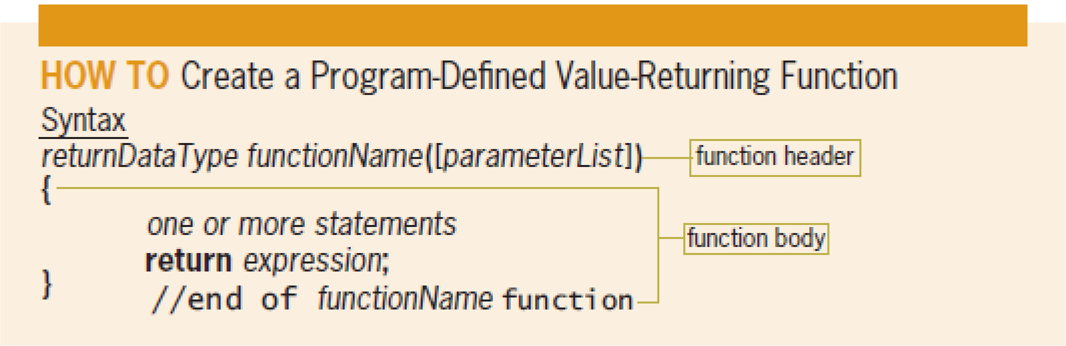
```
**
****
******
********
**********
```

b)

```
* * * * * * * * *
* * * * * * * *
* * * * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
```

Q4. Use a do while loop to accomplish the following. The program should calculate and display the total points and the average the student earned on the tests. The number of tests are not known in advance. When the user enters a –ve number it shows that there are no more marks to enter.

Lab 5: Functions 05th march, 2015

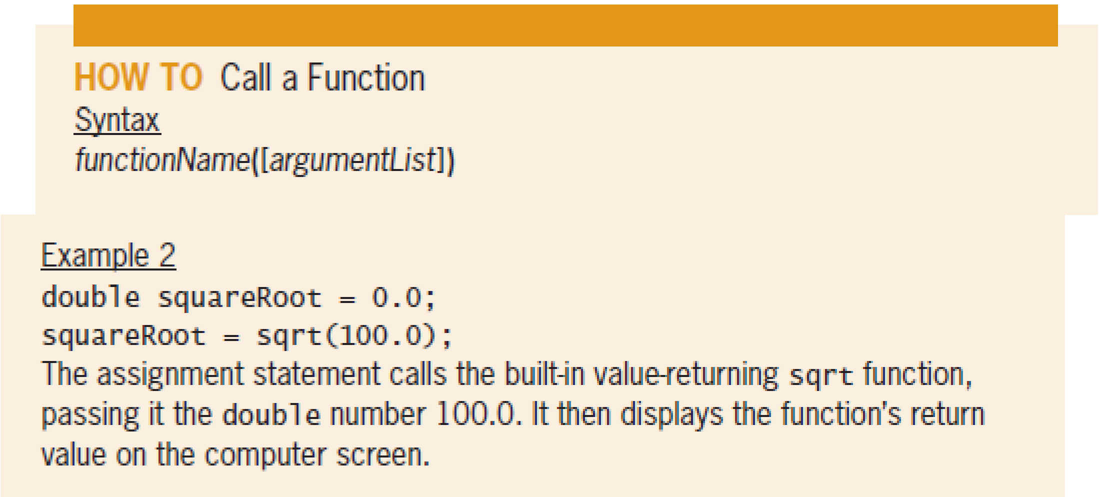


HOW TO Create a Program-Defined Value-Returning Function

Syntax

```
returnDataType functionName([parameterList])
{
        one or more statements
        return expression;
}       //end of functionName function
```

function header
function body



HOW TO Call a Function

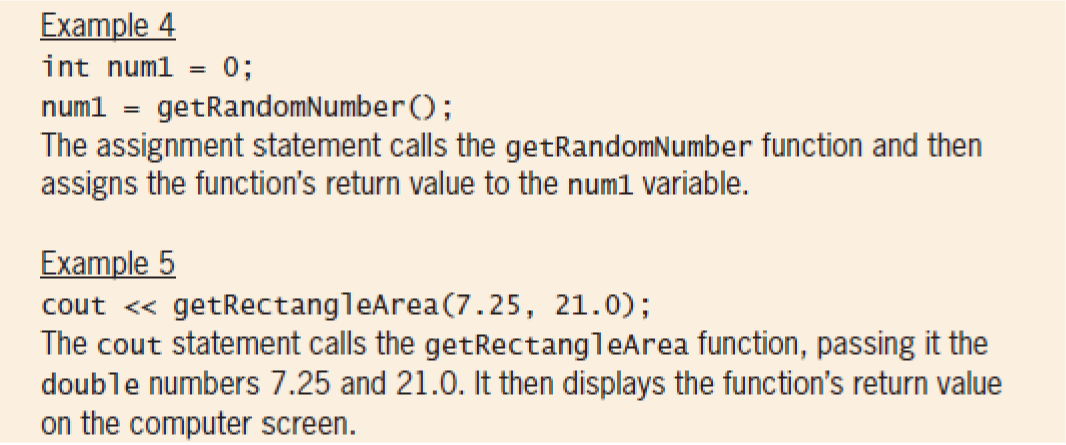
Syntax

functionName([argumentList])

Example 2

```
double squareRoot = 0.0;
squareRoot = sqrt(100.0);
```

The assignment statement calls the built-in value-returning `sqrt` function, passing it the `double` number 100.0. It then displays the function's return value on the computer screen.



Example 4

```
int num1 = 0;
num1 = getRandomNumber();
```

The assignment statement calls the `getRandomNumber` function and then assigns the function's return value to the `num1` variable.

Example 5

```
cout << getRectangleArea(7.25, 21.0);
```

The `cout` statement calls the `getRectangleArea` function, passing it the `double` numbers 7.25 and 21.0. It then displays the function's return value on the computer screen.

Q1) Create a program that allows the user to enter a rectangle's length and width (in feet). The program should calculate and display the rectangle's area in square feet.

Use the following prototype:

doublegetRectangleArea(double len, double wid)
{// add statements here
} //end of getRectangleArea function

The function calculates the area of a rectangle and then returns the result as a double number. Implement a program that calls this function.

Q2) Create a program that allows the user to enter the amount of a salesperson's sales. The program sends these values to a function that calculates a 5% bonus and returns this value to main(). The main program then then displays the bonus on the computer screen. Use the following prototype:

doublegetBonus(int sold, double bonusRate)
{
//add statements here
} //end of getBonus function

Q3) Modify the above program so that it uses four functions

a) to get sales amount from user
b) to get bonus percentage from user
c) to calculate bonus amount
d) to display the total amount of bonus

The calling program should keep on calculating bonuses in this way till the user types N to stop.

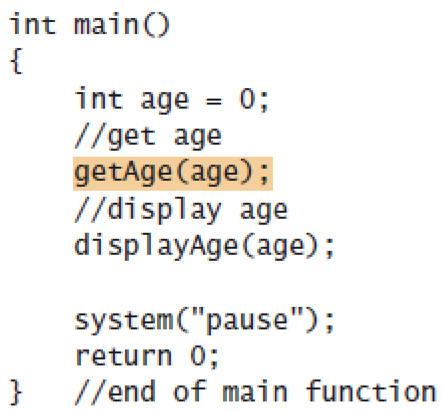
Q4) Create a banking program. Use functions to

a) create an account i.e. take user details(intuserid, double openingbal)
b) credit function
   i) adds a specific amount to the balance
c) debit function
   i) Subtracts a specific amount from the balance
d) display balance

The program should continue to ask for choices till user press 'q' to quit.

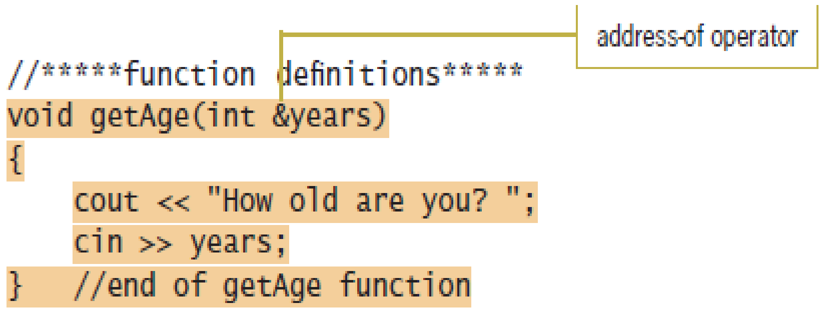
Lab 6: Functions (Pass by reference) 17th march, 2015

Example of Function call of Call by Reference



```
int main()
{
    int age = 0;
    //get age
    getAge(age);
    //display age
    displayAge(age);

    system("pause");
    return 0;
}   //end of main function
```

Example of Function Definition of Call by Reference

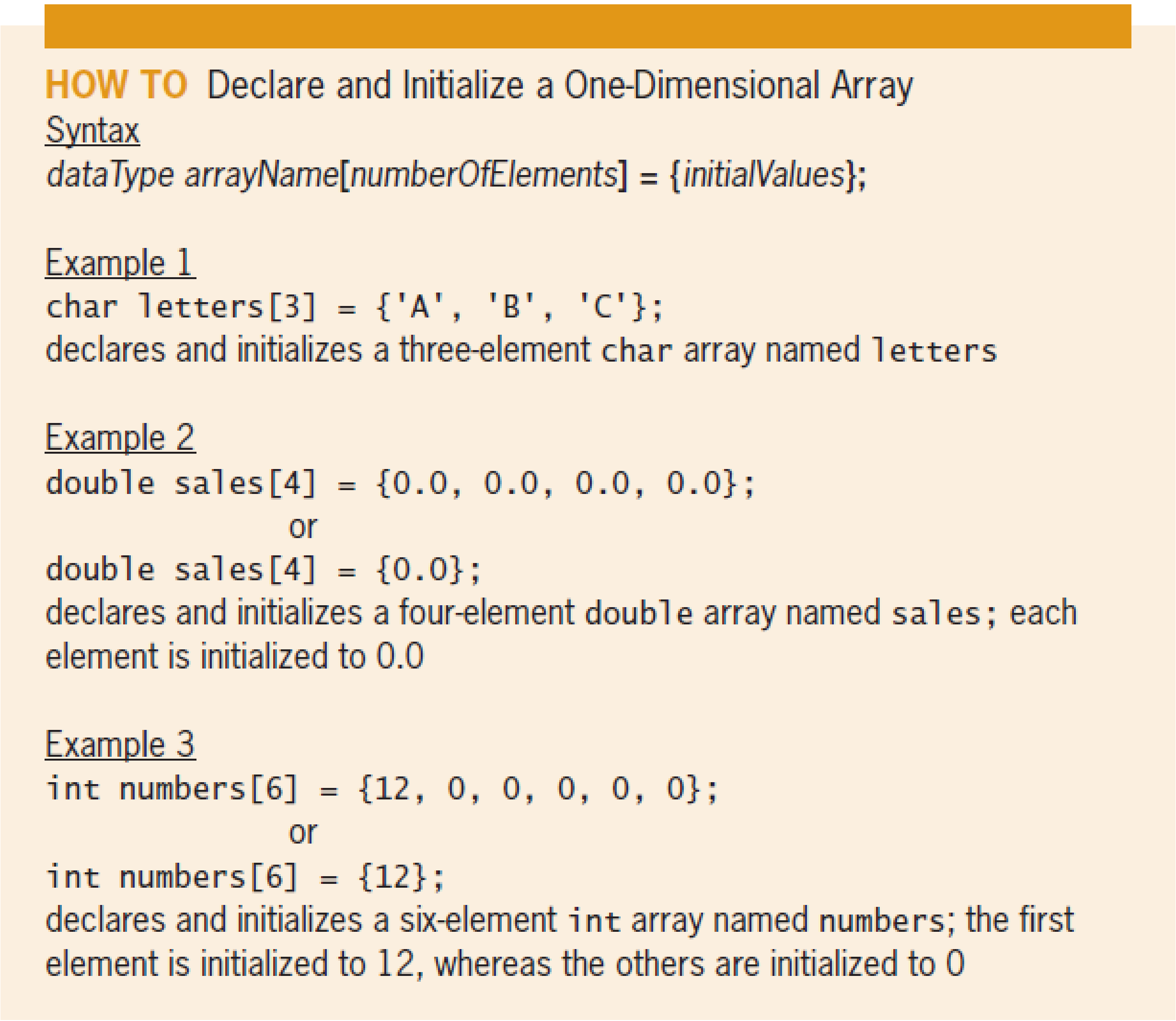


```
//*****function definitions*****
void getAge(int &years)
{
    cout << "How old are you? ";
    cin >> years;
}   //end of getAge function
```

Now implement the following questions through functions with parameters that are passed by reference

1) Write a function that is passed a number num and calculates its square and cube and passes these values back to the main program. Use the following function header. voidCubeNSquare(int&x)
2) Add two time values(hrs1 and minutes1, hrs2 and mins 2) sending the results back to main program as hours and minutes
3) Write a program that will
   - input two floating point numbers : first and second
   - print these values
   - calls function named swap which takes these numbers as input
   - function swap swaps the values of these variables
   - main program prints values again

The main program then prints these results.



HOW TO Declare and Initialize a One-Dimensional Array

Syntax

dataType arrayName[*numberOfElements*] = {*initialValues*};

Example 1

```
char letters[3] = {'A', 'B', 'C'};
```

declares and initializes a three-element `char` array named `letters`

Example 2

```
double sales[4] = {0.0, 0.0, 0.0, 0.0};
```

or

```
double sales[4] = {0.0};
```

declares and initializes a four-element `double` array named `sales`; each element is initialized to 0.0

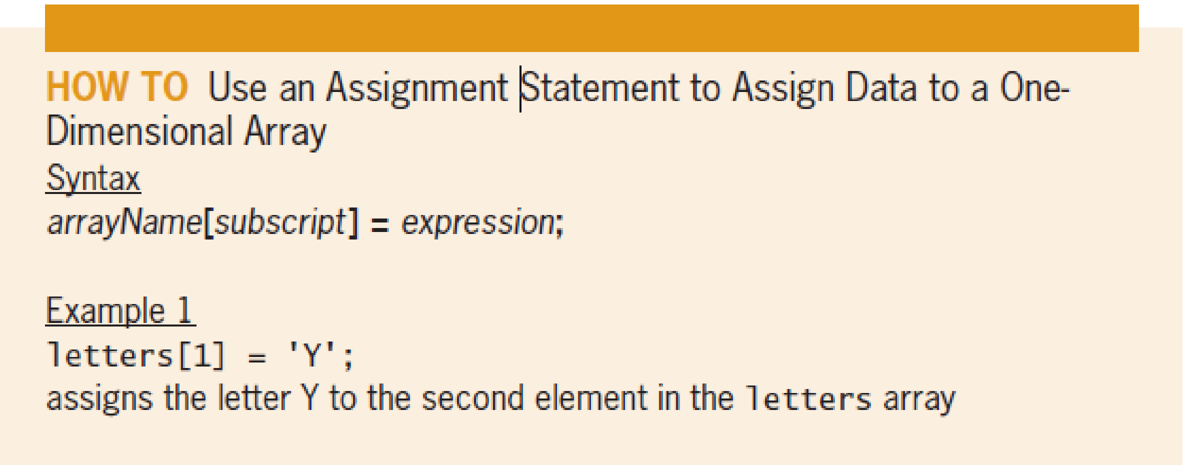
Example 3

```
int numbers[6] = {12, 0, 0, 0, 0, 0};
```

or

```
int numbers[6] = {12};
```

declares and initializes a six-element `int` array named `numbers`; the first element is initialized to 12, whereas the others are initialized to 0



HOW TO Use an Assignment Statement to Assign Data to a One-Dimensional Array

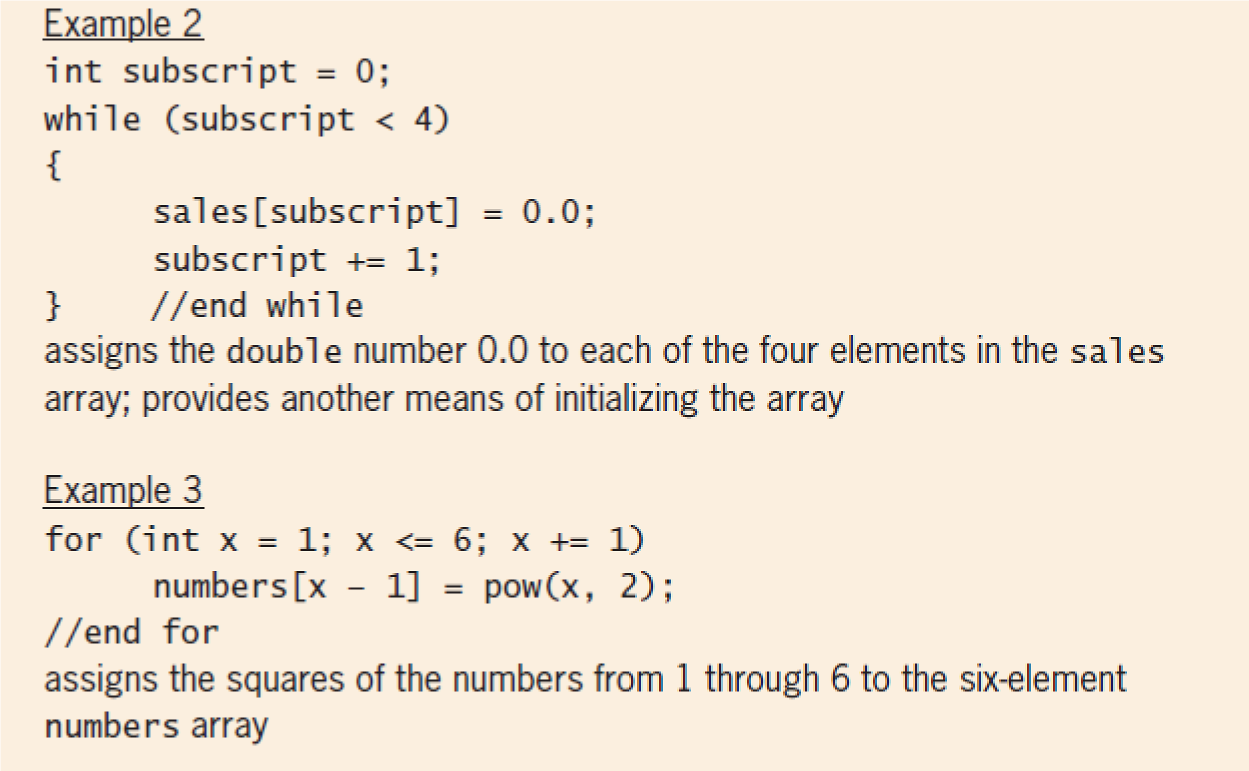
Syntax

arrayName[*subscript*] = *expression*;

Example 1

```
letters[1] = 'Y';
```

assigns the letter Y to the second element in the `letters` array



Example 2

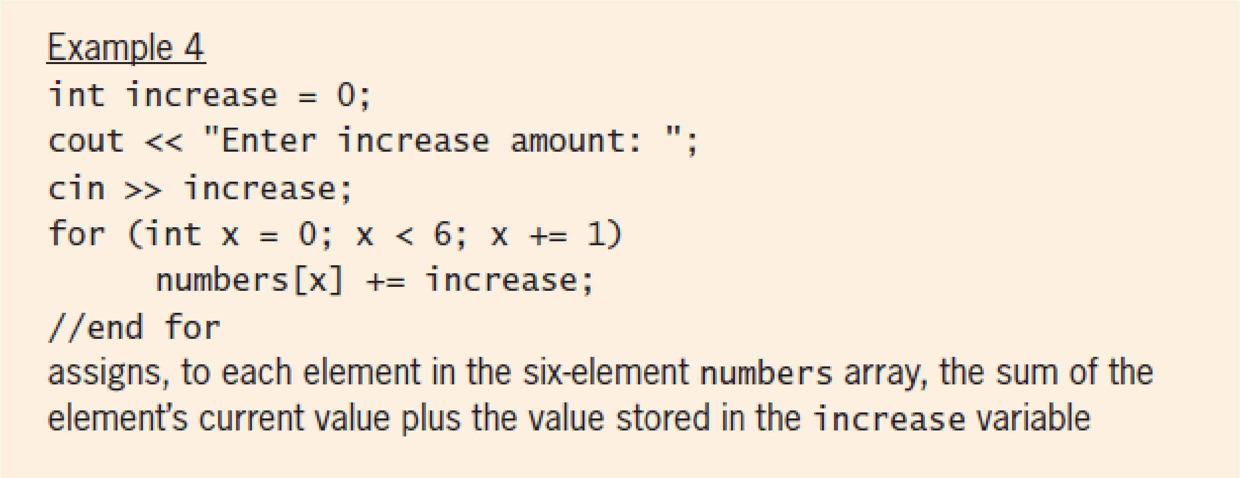
```
int subscript = 0;
while (subscript < 4)
{
      sales[subscript] = 0.0;
      subscript += 1;
}     //end while
```

assigns the double number 0.0 to each of the four elements in the sales array; provides another means of initializing the array

Example 3

```
for (int x = 1; x <= 6; x += 1)
      numbers[x - 1] = pow(x, 2);
//end for
```

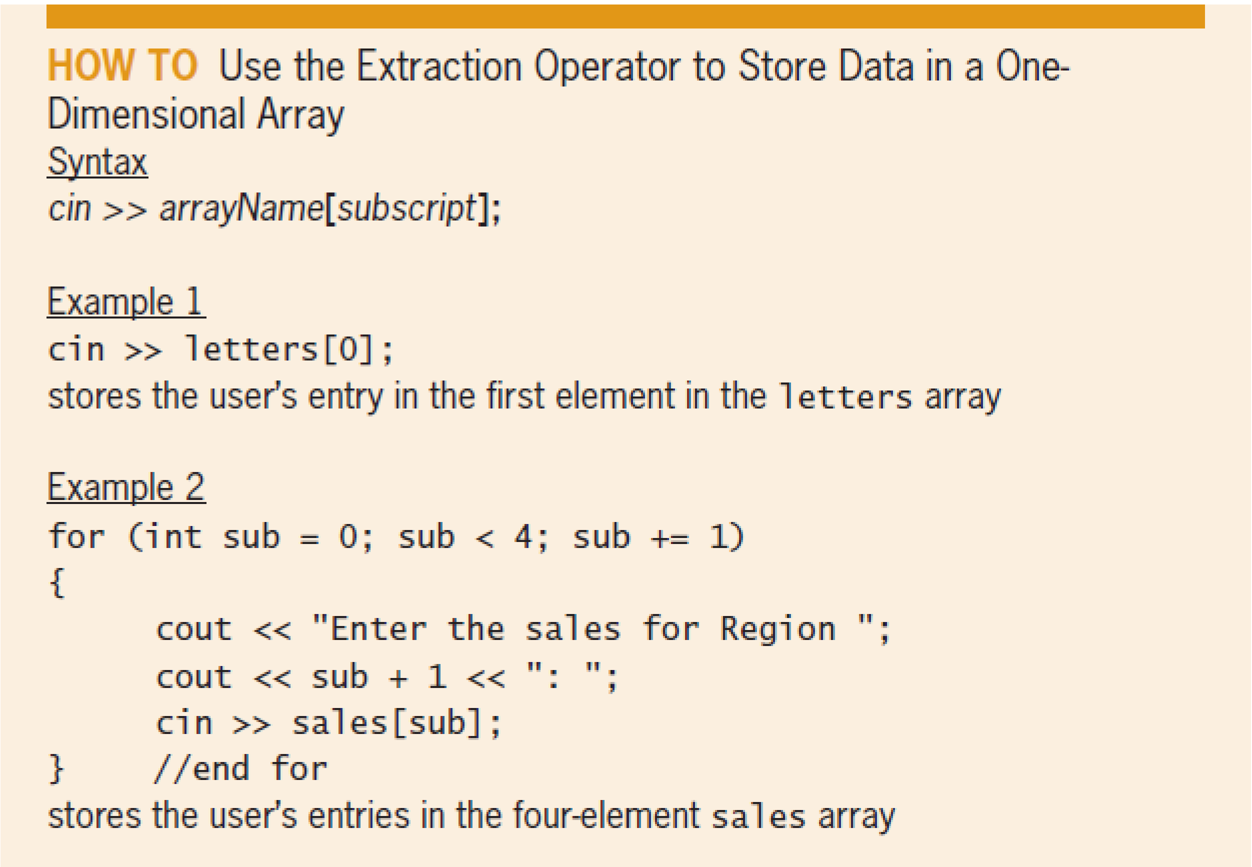
assigns the squares of the numbers from 1 through 6 to the six-element numbers array



Example 4

```
int increase = 0;
cout << "Enter increase amount: ";
cin >> increase;
for (int x = 0; x < 6; x += 1)
      numbers[x] += increase;
//end for
```

assigns, to each element in the six-element numbers array, the sum of the element's current value plus the value stored in the increase variable



HOW TO Use the Extraction Operator to Store Data in a One-Dimensional Array

Syntax

cin >> arrayName[*subscript*];

Example 1

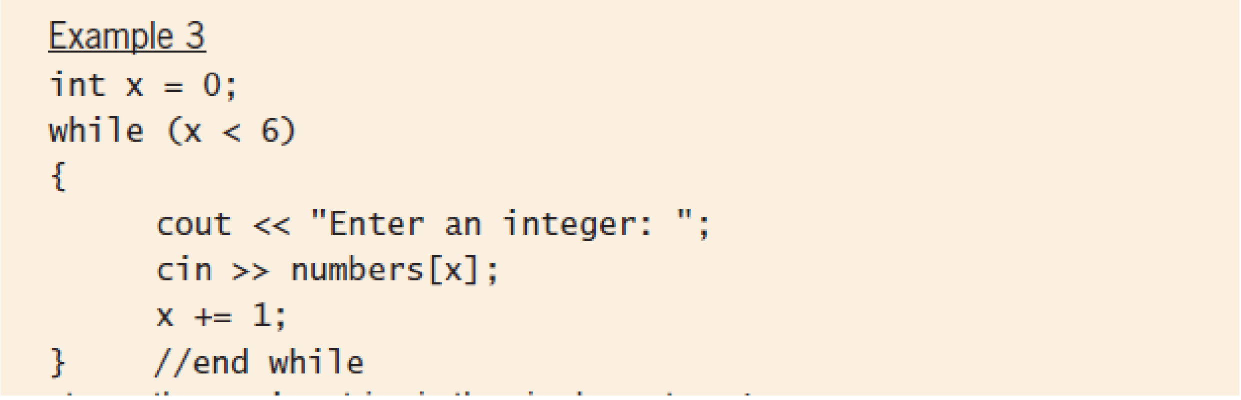
```
cin >> letters[0];
```

stores the user's entry in the first element in the `letters` array

Example 2

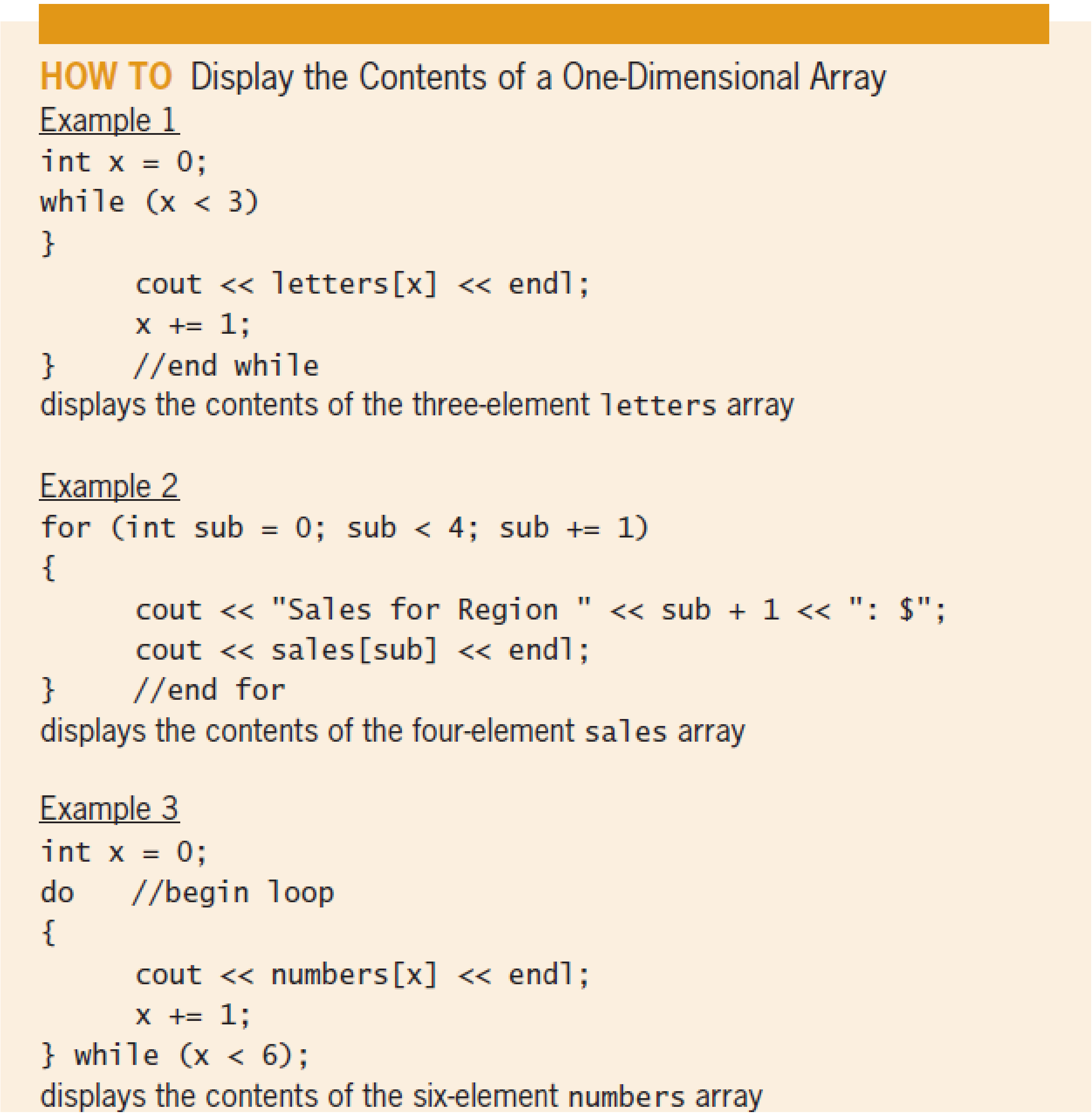
```
for (int sub = 0; sub < 4; sub += 1)
{
      cout << "Enter the sales for Region ";
      cout << sub + 1 << ": ";
      cin >> sales[sub];
}     //end for
```

stores the user's entries in the four-element `sales` array



Example 3

```
int x = 0;
while (x < 6)
{
      cout << "Enter an integer: ";
      cin >> numbers[x];
      x += 1;
}     //end while
```



HOW TO Display the Contents of a One-Dimensional Array

Example 1

```
int x = 0;
while (x < 3)
}
      cout << letters[x] << endl;
      x += 1;
}     //end while
```

displays the contents of the three-element `letters` array

Example 2

```
for (int sub = 0; sub < 4; sub += 1)
{
      cout << "Sales for Region " << sub + 1 << ": $";
      cout << sales[sub] << endl;
}     //end for
```

displays the contents of the four-element `sales` array

Example 3

```
int x = 0;
do    //begin loop
{
      cout << numbers[x] << endl;
      x += 1;
} while (x < 6);
```

displays the contents of the six-element `numbers` array

1. Write C++ statements to do the following:
    a. Declare an array alpha of 15 components of type int.
    b. Output the value of the tenth component of the array alpha.
    c. Set the value of the fifth component of the array alpha to 35.
    d. Set the value of the ninth component of the array alpha to the sum of the sixth and thirteenth components of the array alpha.
    e. Set the value of the fourth component of the array alpha to three times the value of the eighth component minus 57.
    f. Output alpha so that five components per line are printed.
2. What is the output of the following program segment?

```
int temp[5];
for (inti = 0; i< 5; i++)
temp[i] = 2 * i - 3;
for (inti = 0; i< 5; i++)
cout<< temp[i] << " ";
cout<<endl;
temp[0] = temp[4];
temp[4] = temp[1];
temp[2] = temp[3] + temp[0];
for (inti = 0; i< 5; i++)
cout<< temp[i] << " ";
cout<<endl;
```

3. Suppose list is an array of six components of type int. What is stored in list after the following C++ code executes?

```
list[0] = 5;
for (inti = 1; i< 6; i++)
{
list[i] = i * i + 5;
if (i> 2)
list[i] = 2 * list[i] - list[i - 1];
}
```

4. Write a program which reads integers from the user and stores these one by one in an array. User can enter a **maximum** of 100 numbers. Stop taking input when user enters -1. Use a loop to display the values added in the array

5. Write a program that declares a character array Full Name of size 20. Use a loop to enter characters into this array from the user which spell out his full name. Then print the users’ full name from the array. The First and Last name will be separated by a space.

6. Write a program that declares and initializes an array of 10 numbers with values that may be repeated. It then asks the user for a number (key) and checks whether that number exists in the array or not.

7. Write a program that takes as input 10 numbers from the user and assigns then to each array location. Then it adds 5 to each number stored in the array. Then the program

should calculate the sum of these values and also print each number from the array in reverse order (i.e element stored in last location should be printed first).

More Challenging Questions

1. Write a program that declares a character array FullName of size 20. Use a loop to enter characters into this array from the user which spell out his full name. The First and Last name will be separated by a space.Print the user's name such that his surname is printed first and then his First Name

2. Write a program that declared two integer arrays of size 40. It initializes array1 with values 10, 20, 30 and so on using a loop and some formula. It then copies ArrayOne into ArrayTwo.

3. Write a program that declares and initializes an array of 10 numbers with values that may be repeated. The program prints the number of times each number appears in the array i.e the frequency of each number stored.

Lab 8 Character and Strings

Q:Write a program to find the length of string. The user enters a string. The program calculates its length and displays. "You entered a string of length….". You are supposed to do this in two ways:
using a string class function
by writing your own function to calculate the length

Q:Write a program to concatenate one string contents to another. User is prompted to enter two strings. Your program attaches one string to the end of the other. Use character arrays and write your own function for concatenation.

Q:Write a program to count number of words in string. any two words in a sring are separated by a ' ' space. Write a program that inputs a whole sentence and then displays the number of words in that sentence.

Q:Write a program to compare two strings they are exact equal or not. The user enters two strings. These are saved in two character arrays. The program then compares them character by character to see if the strings are same or not.

Q:A palindrome is a string that reads the same in reverse as well. For e.g. abbba is a palindrome. Your program should input a string and display whether the string is a palindrome or not.

Q:Write a program to convert a string in lowercase. Input a string in upper case (Capital letters). Convert each letter to its lower case form in the array. Display the array containing all lowercase letters. You can use built in functions to convert from upper to lower case or you can write your own function as well.

Q:Write a program to find a substring within a string. If found display its starting position. The program inputs two strings and sees if the first string is contained in the second. If yes, it displays iits starting position otherwise prints nort found. For example “ore” and “Lahore” will output 3 whereas “discrete” and “Programming” will output not found.

Q: Write a program to convert a string in uppercase. . Input a string in lower case (small letters). Convert each letter to its upper case form in the array. Display the array containing all uppercase letters. You can use built in functions to convert from lower to upper case or you can write your own function as well.

Q:Create a program that allows the user to enter a NADRA ID card number without the two hyphens. The user’s entry should contain nine characters. If the user did not enter the required number of characters, the program should display the message “The number must contain 9 characters”. Otherwise, the program should insert the two missing hyphens and then display the result on the screen.

Q:Write a program that prompts the user to input a string and outputs the string in uppercase letters. (Use a character array to store the string.)

Q:(Print a string backwards) Write a function stringReverse that takes a character array containing a string as an argument, prints the string backwards and returns nothing. The function should stop processing and return when the terminating null character is encountered.

Q:Write a program that display the date according to instructions. The date should be entered in the following format: mm/yy. Verify that the user entered exactly five characters and that the third character is the slash character (/). If the user did not enter the required number of characters, or if the third character is not a slash, display an appropriate message. Use a sentinel value to end the program.

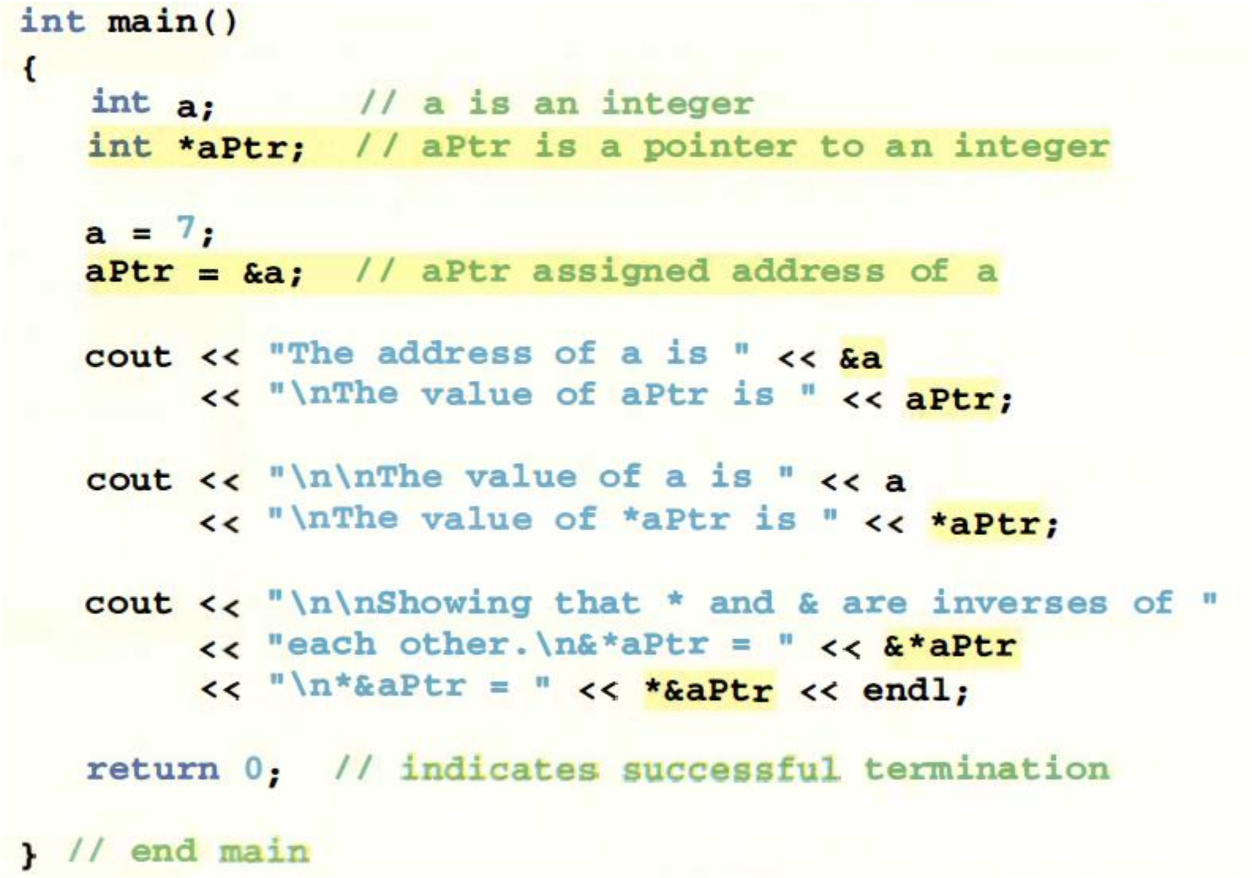
Q:Write a program that display the characters in reverse order. In other words, if the user enters the string “Programming”, the program should display “gnimmargorP”.
Save and then run and test the program.

Q:Write a program that copies arrayA ‘s elements in arrayB at the end.arrayA contains “Programming” and arrayB contains “Structures”. It will results into [Programming Structure]

Q:Create a program that allows the user to enter the name of a company. The program should display the name underlined with a row of hyphens. The number of hyphens should be the same as the number of characters in the company name.

Lab 9: Pointers 5th May, 2015

1. Type the following code and note the outputs.
   Then draw a diagram showing the relevant variables and their values.



```
int main()
{
   int a;         // a is an integer
   int *aPtr;     // aPtr is a pointer to an integer

   a = 7;
   aPtr = &a;     // aPtr assigned address of a

   cout << "The address of a is " << &a
        << "\nThe value of aPtr is " << aPtr;

   cout << "\n\nThe value of a is " << a
        << "\nThe value of *aPtr is " << *aPtr;

   cout << "\n\nShowing that * and & are inverses of "
        << "each other.\n&*aPtr = " << &*aPtr
        << "\n*&aPtr = " << *&aPtr << endl;

   return 0;  // indicates successful termination

} // end main
```

2. Write a program which prints the characters in a string in a reverse order.
   Use the following declarations:
   char s[10] = "abcde";
   char* cptr;
3. For each of the following, write C++ statements that perform the specified task. Assume that Double-precision, floating-point numbers are stored in eight bytes and that the starting address of the array is at location 1002500 in memory. Each part of the exercise should use the results of previous parts where appropriate.
   A. Declare an array of type double called numbers with 10 elements, and initialize the elements to the values 0.0, 1.1, 2.2,. . . , 9.9..Declare a pointer nPtr that points to a variable of type double.
   B. Use a for structure to print the elements of array numbers using array subscript notation.

C. Write two separate statements that each assign the starting address of array numbers to the pointer variable nPtr.

D. Use a for structure to print the elements of array numbers using pointer/offset notation with pointer nPtr.

E. Use a for structure to print the elements of array numbers using pointer/offset notation with the array name as the pointer.

F. Use a for structure to print the elements of array numbers using pointer/subscript notation with pointer nPtr.

G. Refer to the fourth element of array numbers using array subscript notation, pointer/offset notation with the array name as the pointer, pointer subscript notation with nPtr and pointer/offset notation with nPtr.

H. Assuming that nPtr points to the beginning of array numbers , what address is referenced by nPtr + 8? What value is stored at that location?

4. Assume the definitions and initializations:

```
char c = 'T', d = 'S';
char *p1 = &c;
char *p2 = &d;
char *p3;
```

Assume further that the address of c is 6940, the address of d is 9772, and the address of e is 2224. What will be printed when the following statements are executed sequentially?

```
p3 = &d;
cout<< "*p3 = " << *p3 <<endl;   // (1)

p3 = p1;
cout<< "*p3 = " << *p3 ;          // (2)
cout<< ", p3 = " << p3 <<endl;   // (3)

*p1 = *p2;
cout<< "*p1 = " << *p1   ;        // (4)
cout<< ", p1 = " << p1 <<endl;   // (5)
```

Consider the following statements: .5

```
int *p;
inti;
int k;
i = 42;
k = i;
p = &i;
```

After these statements, which of the following statements will change the value of i to 75?

A. k = 75;
B. *k = 75;
C. p = 75;
D. *p = 75;
E. Two or more of the answers will change i to 75.

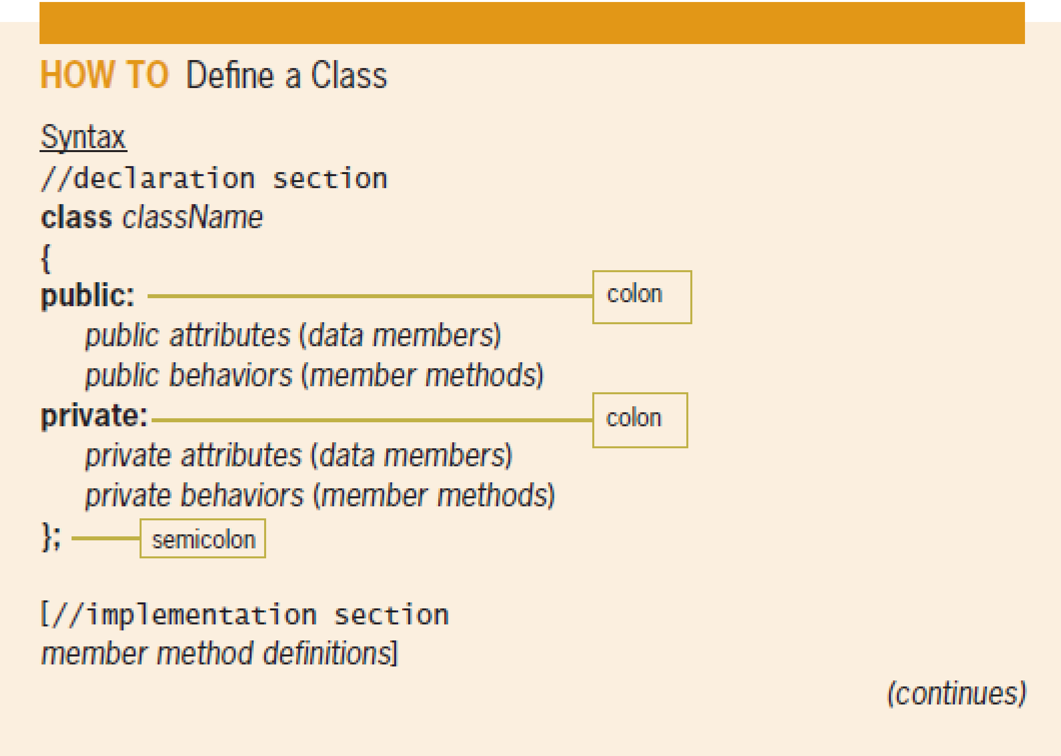
6. Explain the error.

```
char c = 'A';
```

double *p = &c;

7. Give the value of the left-hand side variable in each assignment statement. Assume the lines are executed sequentially. Assume the address of the blocks array is 4434.

```
int main()
{
char blocks[3] = {'A','B','C'};
char *ptr = &blocks[0];
char temp;

temp = blocks[0];
temp = *(blocks + 2);
temp = *(ptr + 1);
temp = *ptr;

ptr = blocks + 1;
temp = *ptr;
temp = *(ptr + 1);

ptr = blocks;
temp = *++ptr;
temp = ++*ptr;
temp = *ptr++;
temp = *ptr;

return 0;
}
```



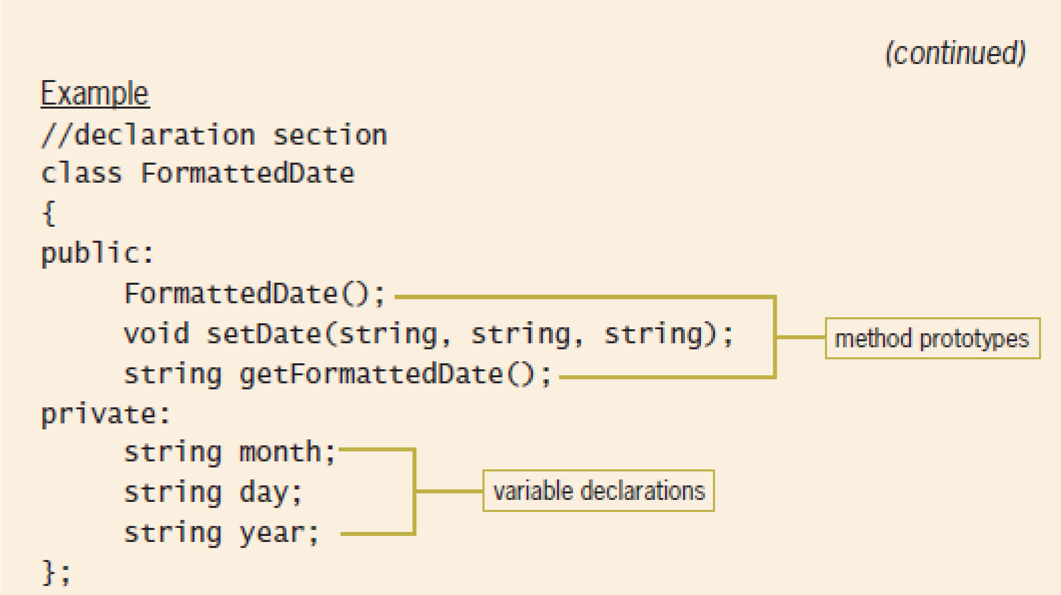
HOW TO Define a Class

Syntax

```
//declaration section
class className
{
public:
    public attributes (data members)
    public behaviors (member methods)
private:
    private attributes (data members)
    private behaviors (member methods)
};

[//implementation section
member method definitions]
```

(continues)

colon
colon
semicolon

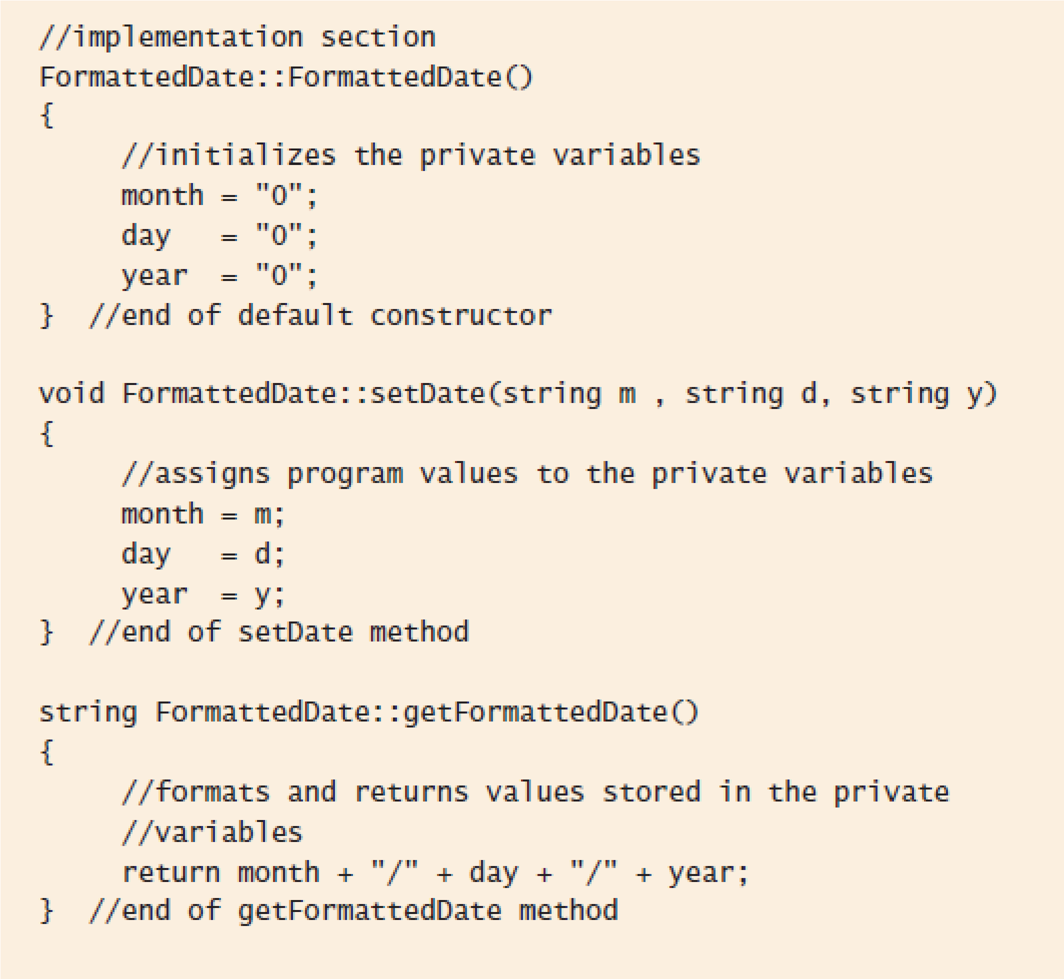


(continued)

Example

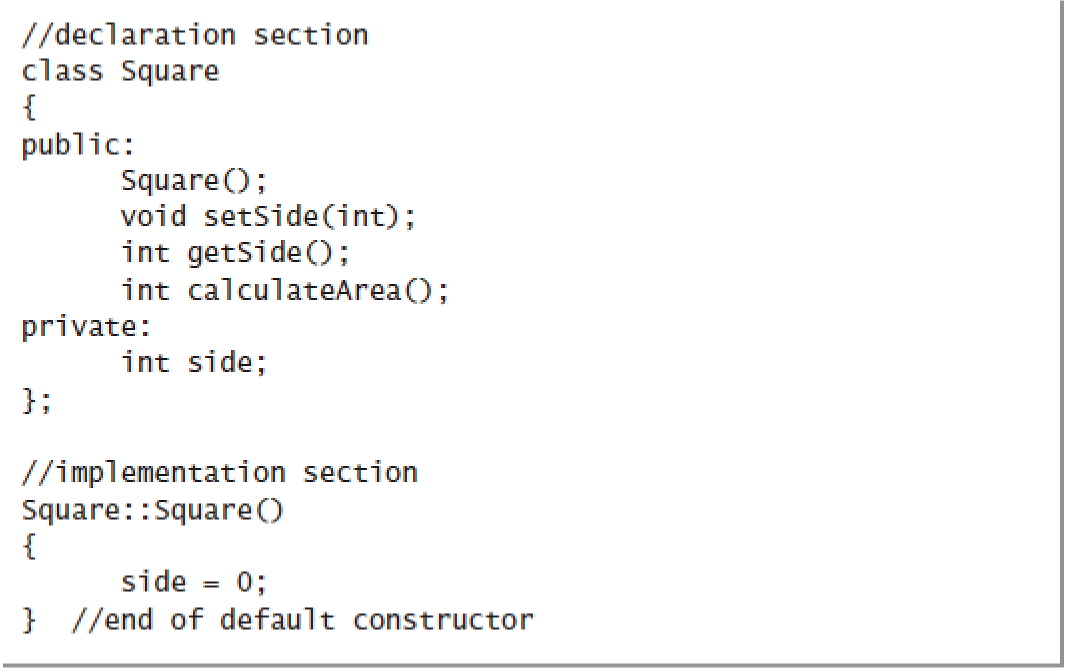
```
//declaration section
class FormattedDate
{
public:
      FormattedDate();
      void setDate(string, string, string);
      string getFormattedDate();
private:
      string month;
      string day;
      string year;
};
```

method prototypes
variable declarations

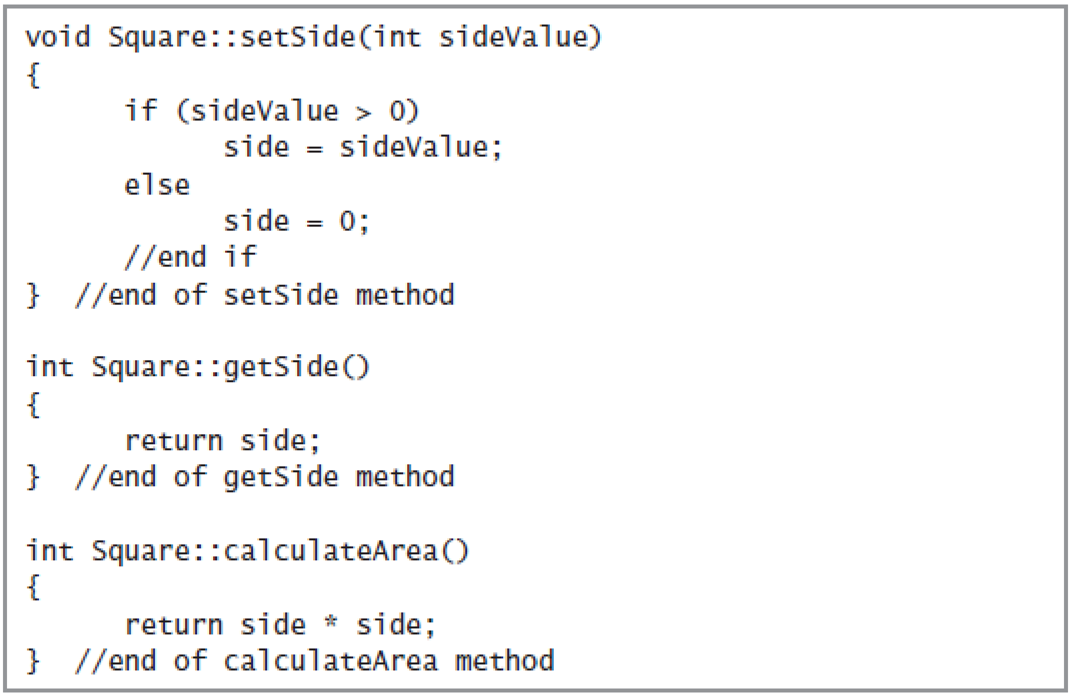


```
//implementation section
FormattedDate::FormattedDate()
{
      //initializes the private variables
      month = "0";
      day   = "0";
      year  = "0";
}  //end of default constructor

void FormattedDate::setDate(string m , string d, string y)
{
      //assigns program values to the private variables
      month = m;
      day   = d;
      year  = y;
}  //end of setDate method

string FormattedDate::getFormattedDate()
{
      //formats and returns values stored in the private
      //variables
      return month + "/" + day + "/" + year;
}  //end of getFormattedDate method
```

8. Type the following code and name the file square.h

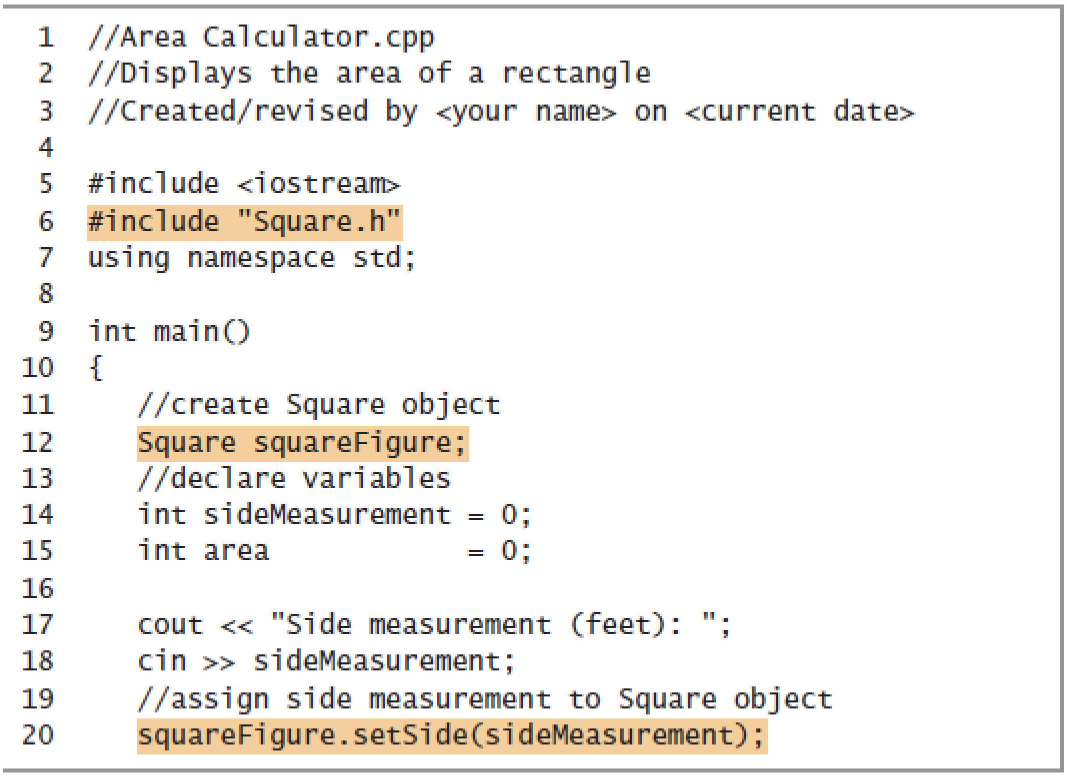


```
//declaration section
class Square
{
public:
       Square();
       void setSide(int);
       int getSide();
       int calculateArea();
private:
       int side;
};

//implementation section
Square::Square()
{
       side = 0;
}  //end of default constructor
```

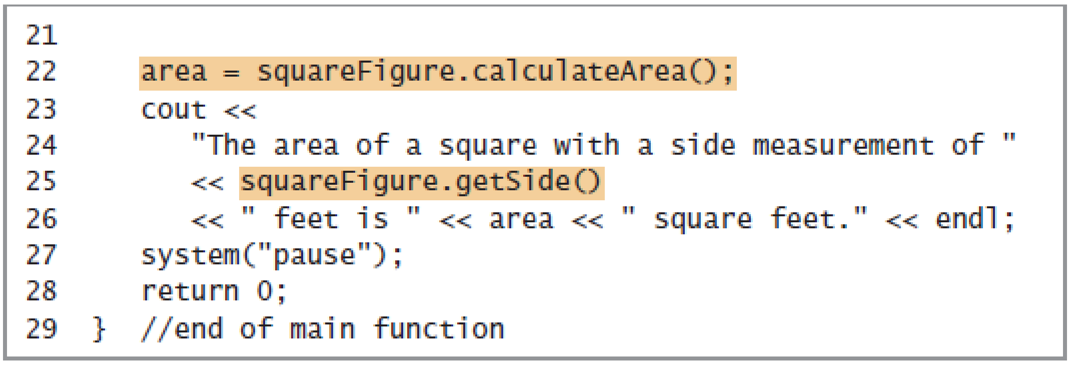


```cpp
void Square::setSide(int sideValue)
{
        if (sideValue > 0)
                side = sideValue;
        else
                side = 0;
        //end if
}  //end of setSide method

int Square::getSide()
{
        return side;
}  //end of getSide method

int Square::calculateArea()
{
        return side * side;
}  //end of calculateArea method
```

Now type the following code in a .cpp file and note the outputs



```cpp
//Area Calculator.cpp
//Displays the area of a rectangle
//Created/revised by <your name> on <current date>

#include <iostream>
#include "Square.h"
using namespace std;

int main()
{
   //create Square object
   Square squareFigure;
   //declare variables
   int sideMeasurement = 0;
   int area            = 0;

   cout << "Side measurement (feet): ";
   cin >> sideMeasurement;
   //assign side measurement to Square object
   squareFigure.setSide(sideMeasurement);
```



```

   area = squareFigure.calculateArea();
   cout <<
      "The area of a square with a side measurement of "
      << squareFigure.getSide()
      << " feet is " << area << " square feet." << endl;
   system("pause");
   return 0;
}  //end of main function
```

9. Write code for a user defined data type class “Time” with private date members hours, minutes and seconds all of which are of integer data type.
    A. Write public get and set functions for each data member i.e. getHour(), get Minutes(), getSeconds() and setHour(), setMinute(), setSecond(). Write a public function display() to diplay the value of time.
    B. Use this class in a program by declaring a variable of type time and set and get values for the hours minutes and seconds.

10. Create a class Rectangle with attributes length and width.
    A. Provide member functions that calculate the perimeter and the area of the rectangle .
    B. Also, provide set and get functions for the length and width attributes. The set functions should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0.

Standard 3-2: There must be adequate support personnel for instruction and maintaining the laboratories.

Lab is managed by a Lab Administrator who is responsible for keeping the hardware and software in working condition. He is also required to ensure that networking of the computers is working properly and Internet is available at each workstation.

The Lab Administrator seeks guidance from the concerned Course Instructor regarding conduct of lab exercises pertaining to different courses. Further, the Lab Administrator is supported in his job function by the Information Technology Resource Center Staff located.

Standard 3-3: The University computing infrastructure and facilities must be adequate to support program’s objectives.

The facilities mentioned in the above labs are adequate to support the objectives of the B.Sc. (Hons) in Software Engineering program. Students of this program who are residing in the University Hostel have been provided computers which are equipped with necessary software along with Internet access.

CRITERION 4: SUPPORT AND ADVISING

Although classes start at 6pm but permanent faculty and the administration staff is available from 1pm onwards. Visiting faculty may also be called in based on the students' requests. Any extra support in terms of course content and lab practice is arranged by the department.

Standard 4-1: Courses must be offered with sufficient frequency and number for students to complete the program in a timely manner.

The B.Sc. (Hons) in Software Engineering program comprises of forty two (42) courses spread over five years (10 semesters) of part time study.

In each semester, normally three courses are offered which constitute a study load of 9 to 12 credit hours. Each course in the B.Sc. (Hons) in Software Engineering program is offered once in an academic year, either in the Spring, Summer, or Fall semester. Students can register for 6 to 8 credit hours in Summer semester.

Elective courses are offered depending upon the availability of the Instructor and the interest of the students.

The students are encouraged to take up elective courses from other Schools / Departments of the University. The students have availability of courses to choose from other departments to satisfy their elective requirements.

Standard 4-2: Courses in the major area of study must be structured to ensure effective interaction between students, faculty and teaching assistants.

EFFECTIVE FACULTY / STUDENT INTERACTION

There is a strong interaction between Course Instructor and the students during the conduct of the course. Students are free to ask any relevant questions from the Instructor during the class as well as after class hours. Student can also communicate with the Instructor through electronic mail or social media whatever mode is preferred by the instructor.

Standard 4-3: Guidance on how to complete the program must be available to all the students and access to academic advising must be available to make course decisions and career choices.

- The prospectus of the University is published every year and contains detailed information about the program. Along with study plan for each semester. Student's queries are also addressed in Orientation Session organized before the start of academic year by the School of Computer & Information Technology.

- The faculty member of the School along with the Dean are available to provide guidance and counseling relating to all academic matters, as and when required. Students are free to discuss their academic and personal problems with the Dean, Faculty and Coordinator of the School.

- Every possible effort is made to satisfy the student's queries and provide solutions to their problems.

- Most Visiting Faculty Members are experienced professionals and the students have opportunity to discuss with them their queries regarding academic and professional matters.

- By means of departmental bulletin board, students have updated information about seminars, workshops, conferences and other co-curricular events in the field of Computer Science and Software Engineering.

CRITERION 5: PROCESS CONTROL

- Study Plan for the degrees offered at SCIT are reviewed every year through board of studies meeting held anually.

- Course progress meetings are conducted during the semester time to check if the course content is being covered as planned.

- Exam papers are evaluated for quality to ensure that the evaluation tools are being designed as per standards.

- Instructors and students are encouraged to share their experiences with the department through meetings or through emails.

Standard 5-1: The process by which students are admitted to the program must be based on quantitative and qualitative criteria and clearly documented. This process must be periodically evaluated to ensure that it is meeting its objectives.

- **PROGRAM ADMISSION CRITERIA**

  Applicants who have passed Intermediate(with either Mathematics or Computer Science) in minimum 2nd division are eligible to apply to the B.Sc. (Hons) in Software Engineering.

  As part of the admission process, all the applicants are required to take an Admission test and appear in an interview.

- **PROGRAM/CREDIT TRANSFER**

The School refers all transfer cases to the University Equivalence Committee. The Equivalence Committee, after thorough scrutiny in light of the HEC guidelines, gives approval for all transfers.

- **EVALUATION OF ADMISSION CRITERIA**

  The admission criterion is reviewed annually in light of the HEC guidelines. The Board of Studies meets every year and reviews all matters regarding the program. In addition Academic Council of the University also reviews the Admission procedure and subsequent approval is taken from the Board of Governors of the University.

Standard 5-2: The process by which students are registered in the program and monitoring of students progress to ensure timely completion of the program must be documented. This process must be periodically evaluated to ensure that it is meeting its objectives.

- **PROCESS OF REGISTRATION**

  The process of registration being followed at the School of Computer and Information Technology is a two-pronged process. The coordinator under the supervision of the Dean of the School keeps a record of the student registration in courses. This record is then passed on to the Registrar of the University and the Examination & Quality Assurance departments.

- **MONITORING STUDENTS PROGRESS**

  - The student progress is carefully monitored throughout their academic stay at the School. The program follows continuous assessment procedures.
  - The results of the students are carefully recorded and monitored by the School and passed on to the Examination and Quality Assurance department.
  - The faculty, Head of Department and the Dean meet on a regular basis to discuss all student related issues.
  - Attendance records, class performance records of all students are also maintained by the School.
  - Transcripts are prepared by the examination department at the end of every semester.
  - These transcripts are mailed to the students at the end of the semester.

- **EVALUATION AND IMPROVEMENT**

  The process is evaluated by conducting periodical peer reviews.

Standard 5-3: The process of recruiting and retaining highly qualified faculty members must be in place and clearly documented. Also processes and procedures for faculty evaluation, promotion must be consistent with institution mission statement. These processes must be periodically evaluated to ensure that it is meeting with its objectives.

- **FACULTY RECRUITEMENT PROCESS**

- The School of Computer and Information Technology follows a thorough process for the recruitment of faculty in line with the BNU and HEC guidelines.
- The process begins with identification of faculty (preferably foreign qualified).
- They are then invited to give mock lectures, which are attended by the Dean and Permanent Faculty Members of the School.
- Based on the mock lecture, the School of Computer & Information Technology proposes their name to University HR Department so that the formal recruitment process may begin.
- These cases are then put before the Selection Board that interviews the candidates. On the recommendation of the Selection, the Board of Governors of BNU give the final approval.

Standard 5-4: The process and procedures used to ensure that teaching and delivery of course material to the students emphasizes active learning and that course learning outcomes are met. The process must be periodically evaluated to ensure that it is meeting its objectives.

- Before the start of each semester, faculty is briefed about all procedures and guidelines are shared for preparing the course outlines, designing exams, and maintaining the course logs for HEC authorities.

- Course outline template is also shared with faculty to ensure that leaning outcomes are clearly mentioned and minimum number of class tests and assignments are conducted in each course. The fulfillment of pre requisites is also verified. The course descriptions along with recommended text books are also revised every year.

- The process is reviewed every summer to identify further improvements in the processes. Faculty feedback is also considered important and steps are taken to incorporate any suggestions by faculty or program teams.

Standard 5-5: The process that ensures that graduates have completed the requirements of the program must be based on standards, effective and clearly documented procedures. This process must be periodically evaluated to ensure that it is meeting its objectives.

- The Administrative Coordinator maintains complete records of the students. These records are reviewed at the start and end of every semester to ensure the student is progressing and meeting all requirements of the program.
- The Registrar office and Quality Assurance department maintain files on each student. These files contain past and ongoing academic record of the students. At the end of each semester these records are reviewed as a means to check student performance.
- At the time of graduation the record of each student is thoroughly scrutinized to ensure that the student has fulfilled all requirements of the degree program.

- After ensuring that all requirements have been met, the student is considered eligible to graduate.

CRITERION 6: FACULTY

Faculty members must be current and active in their discipline and have the necessary technical depth and breadth to support the program. There must be enough faculty members to provide continuity and stability, to cover the curriculum adequately and effectively, and to allow for scholarly activities. To meet this criterion the standards in this section must be satisfied.

Standard 6-1: There must be enough full time faculties who are committed to the program to provide adequate coverage of the program areas/ courses with continuity and stability. The interests and qualifications of all faculty members must be sufficient to teach all coursed, plan, modify and update coursed and curricula. All faculty members must have a level of competence that would normally be obtained through graduate work in the discipline. The majority of the faculty must hold a Ph. D. in the discipline.

- Complete the following table indicating program areas and number of faculty in each area.

Sr.#	Course Code	Course Tiltle	Cr. Hrs	Teacher	Highest Degree
1	MTH-105	Applied Physics	3	Salim Butt	MS in Enggineering
2	MTH-101	Calculus and Analytical Geometry-I	3	Ms. Samra Abbas	MS in maths
3	SE-101-A	Communication Skills - I	3	Uzma Touqir	MS in english
4	SE-101-B	Communication Skills-II	3	Ms. Uzma Tauqeer	MS in English
5	CSC-302	Computer Architecture	3	Dr. M Kamran	PHD electronics
6	CSC-205	Computer Org. and Assembly Language	4	Rafay Chughtai	MS in Enggineering
7	CSC-210	Data Communication and Networks	3	Dr. Kamran	PHD in Electronics
8	CSC-409	Data Mining and Warehousing	3	Touqir Rana / Tehmina	MS in CS
9	CSC-204	Data Structures	4	Mr. Nouman Ali	MS in CS
10	CSC-211	Database Management Systems	4	Mr. Asim Tanvir	MS in IT
11	CSC-104	Digital Logic Design	4	Dr. M Kamran	PHD electronics
12	CSC-105	Discrete Structures	3	Ms. Sameen Raza	MS in CS
13	CSC-107	Fundamentals of Computers	4	Mr. Sheraz Parvez	MS in CS
15	CSC-305	Human Computer Interaction	3	Ms. Madiha Ijaz	MS in CS
16	CSC-108	Introduction to Programming	4	Mr. Zeyad Amin	MS in CS

17	CSC-213	Object Oriented Programming	4	Nouman Ali Shah	MS in CS
18	CSC-313	Object Oriented Analysis and Design	3	Nouman Ali Shah	MS in CS
19	CSC-403	Operating Systems	3	Nouman Ali Shah	MS in CS
20	HUM-202	Pakistan and Islamic Studies	3	Atta-ur-Rehman	MA
21	PRJ-401	Project-I	3	Dr. K-Zia	PHD in Electronics
23	HUM-401	Research and Professional Issues	3	Dr. Qayyum	PHD in Stats
24	CSC-316	Software Construction	4	Mr. Zeyad Amin	MS in CS
25	CSC-412	Software Project Management	3	Ms. Mahrukh	MS in CS
26	CSC-317	Software Requirement Engineering	4	Natah Ali Mian	MS in CS
27	CSC-314	Study of Algorithms	3	Dr. Kamran	PHD in Electronics

Table 4.6: Faculty Distribution by Program Area

- **FACULTY RESUMES**

Standard 6-2: All faculty members must remain current in the discipline and sufficient time must be provided for scholarly activities and professional development. Also, effective programs for faculty development must be in place.

- All faculty members in the School of Computer and Information Technology should have a Master’s Degree from foreign or local university. In addition they are current in their area of expertise and have taught the courses allocated to them previously as well.

- Full time faculty members are assigned a maximum load of three courses which entails 9 to 12 semester credit hour of student contact. Keeping in view this load the fulltime faculty has sufficient time for professional development. Furthermore, the fulltime faculty is not given any teaching assignments in summer and they can fully devote their summer time for professional development.

- Faculty is encouraged to participate in seminars, workshops and conferences in the area of their interest.

Standard 6-3: All faculty members should be motivated and have job satisfaction to excel in their profession.

- The faculty member is provided a congenial working environment which is conducive for teaching and research. Air-conditioned offices workstations with internet connectivity and access to digital library are standard features of the faculty working environment.

- Faculty members can purchase any book of their choice without hindrance. Faculty can also undertake professional development training and also get leave for improving their qualification at any other Institution, subject to providing a service bond.

- The performance of faculty is appraised on annual basis and they are awarded annual increment based on the appraisal.

- All the above features help in motivating the faculty in their job.

- Survey of faculty is conducted annually (on HEC approved Proforma # 5) in which the faculty provides its input on work environment and their own performance during the year.

- The survey is quite effective in faculty assessing, the views of the faculty for improving the work environment and facilities.

CRITERION 7: INSTITUTIONAL FACILITIES

Institutional facilities, including library, classrooms and offices must be adequate to support the objective of the program. To satisfy this criterion a number of standards must be met.

Standard 7-1: The institution must have the infrastructure to support new trends in learning such as e-learning.

- ***Please refer to Criterion 6***

Standard 7-2: The library must possess an up-to-date technical collection relevant to the program and must be adequately staffed with professional personnel.

Professional Development

The librarians have been trained in MARC records development and cataloging in a new integrated Library System (ILS). Further, training in the use of the software has been given. Any Archives and Records Management Course for all librarians and representatives of each university department have been trained. The need for this has arisen as a new Archives and a Records Management program has been initiated at the University.

Collection Development

A collection policy has been formulated to guide the library in its development of the collections (see Appendix A)

Library Committee

The BNU Library is guided by the Library Committee for effective management. Dean, Heads of schools are members and library liaisons are nominated from all departments.

Annual Report

The Chief Librarian prepares an annual report to present to the Vice Chancellor of the University, highlighting the accomplishment, problems and needs of the library. Utilization of resources and statistical data is presented in this report.

Library Budget

Annual Budget of BNU Library is Rs. 5.9 million

List of All Materials in BNU Library

Sr.#	Name of Item	Quantity 2015
1	Books	13224
2	Photocopies of Books	66
3	Downloaded E-Books	3000
4	DVD's (movies for TFT)	1053
5	Downloaded Movies (for TFT)	725
6	VHS's	626
7	Art Catalogues	1173
8	Theses	593
9	Reports	2070

Journals / Magazine , Newspapers

Sr. No	Name of Item	Quantity
1	Journals / Magazines (Subscribed)	35
	Journals / Magazines (Complementary)	60
2	Daily Newspapers	13

Online Resources

Sr. No	Name of Source	Availability
1	EVERGREEN OPAC	Online
2	HEC Digital Library	Online
3	ARTSTOR	Online
4	JSOTR	Online

Library Staff

Sr. No	Campuses / Library	No. of Staff Members
1	City Campus	1
2	New Campus Library	6
	Total Staff Members	**7**

BNU Library URL

http://WWW.bnu.edu.pk/index.php?otion=com_content&view=article&id=165&ltemid=484

Library Membership

Membership:	2236
Faculty:	0263
Students:	1895
Staff:	0078

Standard 7-3: Class-rooms must be adequately equipped and offices must be adequate to enable faculty to carry out their responsibilities.

- **CLASSROOMS:**
  - All the classrooms in the School of Computer and Information technology are air-conditioned and have multimedia projector / LCD screens to help in the teaching / learning process.
  - The average class size is 25 students so that instruction can be imparted to students in an effective manner

FACULTY OFFICES:

- ***Please refer to Standard 6- 3***

CRITERION 8: INSTITUTIONAL FACILITIES

The institution's support and the financial resources for the program must be sufficient to provide an environment in which the program can achieve its objectives and retain its strength.

Facility	Description
Land	The total land area of Beaconhouse National University's New Campus is 33 acres.
Buildings	The built-up area of the Beaconhouse National University New Campus is 322,000 sqft. In Phase – I, the New Campus has three academic blocks, one central block and one administration block.

	The first academic block comprising 107,000 sqft areas is operational at the New Campus. The second academic block comprising 56,000 sqft areas is scheduled to start its operation in September, 2011. The remaining buildings are at different stages of construction.
Roads network & Parking	BNU has an internal road network of 1.5 Km. This road links different academic and administrative buildings. Walkways on the sides of the roads have been constructed for easy movement of students and staff. Fire hydrants at different points along the road have also been provided. The New Campus in phase – I has parking space for 400 vehicles. The adjoining areas of the campus can accommodate more than 600 vehicles.
Lawns & Open Spaces	BNU is an environment friendly organization. In the campus design phase special attention has been paid to maintaining bio-diversity of the area. More than 50 % of the campus spaces have been left open and green. Each of the academic and other blocks has a lawn attached to it and is equally used by student, faculty and staff for academic and recreational purposes. The total cost of the planned landscape is Rs.10 m.
ICT	BNU's focus on information and communication technologies is evident from the 1800 nodes system planned for the campus. Already 600 nodes are active providing the users internet connection and IP telephony facility. This back bone is also meant for IP surveillance and access control systems for the buildings.
Sports facilities	Opportunities to participate in sports and extra-curricular events at BNU exit. The University already has set up different indoor and outdoor sports facilities for students. A football field with dimensions of 180 ft x 330 ft is available. This facility also has a cricket turf for hard ball matches. The university has also set up badminton court and table tennis play areas for students.
Canteen	BNU is making significant investment in setting up a four floor purpose built cafeteria for its students, faculty and staff. Work on the structure is being carried out these days. Once completed this facility will provide dining facility to students, staff and faculty. The lower ground floor will comprise an executive dining hall for faculty and senior staff of the university. The ground floor would comprise of a restaurant area offering variety of foods and drinks. The first floor of the cafeteria would be reserved for female students and contain a common room and a prayer area. The top floor of the cafeteria would include separate gyms and work out areas for male and female students. The new canteen would provide campus community the opportunities to find some time to relax and enjoy in free time. Like other campus areas, the cafeteria would have Wi-Fi facilities on all floors.
Furniture	Ergonomically designed furniture has been planned across the campus. Services of design firms have been hired to meet the requirements for studios and classrooms.

Standard 8-1: There must be sufficient support and financial resources to attract and retain high quality faculty and provide the means for them to maintain competence as teachers and scholars.

- The faculty of School of Computer and Information Technology market based salaries along with standard service benefits i.e. Provident Fund, Annual Leave, Medical Leave, and Medical Insurance.

- The Institute has sufficient budgeted fund to support the faculty. The Institution also has funds to support faculty needs for teaching and research purposes.

- The School of Computer and Information Technology has three Coordinators to handle all Administrative and Coordination tasks, so that the faculty is free to concentrate on teaching and research.

Standard 8-2: There must be an adequate number of high quality graduate students, research assistants and Ph. D. students.

- The School of Computer and Information Technology does not have Master Program therefore, there are no graduate students.

Standard 8-3: Financial resources must be provided to acquire and maintain Library holdings, laboratories and computing facilities.

- **LIBRARY**
  - ***Please refer to Standard 7- 2***
- **LABORATORY**
  - ***Please refer to Criterion 3***
- **COMPUTING FACILITIES**
  - ***Please refer to Criterion 3***

Self Assessment Report

(Rubric Form)

School of Computer & Information Technology

Prepared by: Program Team of SCIT

Presented by: Quality Assurance Department

Criterion 1 - Program Mission, Objectives and Outcomes	Weight = 0.05				
	Score				
	5	**4**	**3**	**2**	**1**
Does the program have documented outcomes for graduating students?	5				
Do these outcomes support the program objectives?		4			
Are the graduating students capable of performing these outcomes?		4			
Does the department assess its overall performance periodically using quantifiable measures?			3		
Is the result of the program assessment documented?			3		
Total Encircled Value (TV)	**19**				
Score 1 (S1) = {TV / (No. of Questions * 5)} * 100 * Weight	**3.80**				

Criterion 2 - Curriculum Design and Organization	Weight = 0.20				
	Score				
	5	**4**	**3**	**2**	**1**
Is the curriculum consistent?		4			
Does the curriculum support the program's documented objectives?	5				
Are theoretical background, problem analysis and solution design stressed within the program's core material	5				
Does the curriculum satisfy the core requirements laid down by respective accreditation bodies? (Refer to appendix A of the Self Assessment Report Manual)	5				
Does the curriculum satisfy the major requirements laid down by HEC and the respective councils / accreditation bodies? (Refer to appendix A of Self Assessment Manual)	5				
Does the curriculum satisfy the general education, arts and professional and other discipline requirements as laid down by the respective body / councils? (Refer to appendix A of Self Assessment Manual)		4			
Is the information technology component integrated throughout the program?	5				
Are oral and written skills of the students developed and applied in the program?	5				
Total Encircled Value (TV)	38				
Score 2 (S2) = {TV / (No. of Questions * 5)} * 100 * Weight	**19.00**				

Criterion 3 - Laboratories and Computing Facilities	**Weight = 0.10**				
	Score				
	5	**4**	**3**	**2**	**1**
Are laboratory manuals / documentation / instructions etc. for experiments available and ready accessible of faculty and students?					1
Are there adequate number of support personnel for instruction and maintaining the laboratories?				2	
Are the University's infrastructure and facilities adequate to support the program's objectives?				2	
Total Encircled Value (TV)	**5**				
Score 3 (S3) = {TV / (No. of Questions * 5)} * 100 * Weight	**3.33**				

Criterion 4 - Student Support and Advising	**Weight = 0.10**				
	Score				
	5	**4**	**3**	**2**	**1**
Are the courses being offered in sufficient frequency and number for the students to complete the program in a timely manner?		4			
Are the courses in the major area structured to optimize interaction between the students, faculty and teaching assistants?		4			
Does the University provide academic advising on course decisions and career choices to all students?				2	
Total Encircled Value (TV)	**10**				
Score 4 (S4) = {TV / (No. of Questions * 5)} * 100 * Weight	**6.67**				

Criterion 5 - Process Control	**Weight = 0.15**				
	Score				
	5	**4**	**3**	**2**	**1**
Is the process to enroll students to a program based on quantitative and qualitative criteria?			3		
Is the process above clearly documented and periodically evaluated to ensure that it is meeting its objectives?			3		
Is the process to register students in the program and monitoring their progress documented?			3		
Is the process above periodically evaluated to ensure that it is meeting its objectives?			3		
Is the process to recruit and retain faculty in place and documented?		4			

Are the processes for faculty evolution & promotion consistent with the institution mission?		4			
Are the processes in 5 and 6 above periodically evaluated to ensure that they are meeting their objectives?			3		
Do the processes and procedures ensure that teaching and delivery of course material emphasize active learning and that course learning outcomes are met?	5				
Is the process in 8 above periodically evaluated to ensure that it is meeting its objectives?	5				
Is the process to ensure that graduates have completed the requirements of the program based on standards and documented procedures?		4			
Is the process in 10 above periodically evaluated to ensure that it is meeting its objectives?			3		
Total Encircled Value (TV)	**40**				
Score 5 (S5) = {TV / (No. of Questions * 5)} * 100 * Weight	**10.91**				

Criterion 6 - Faculty	**Weight = 0.20**				
	Score				
	5	**4**	**3**	**2**	**1**
Are there enough full time faculty members to provide adequate coverage of the program areas / courses with continuity and stability?					1
Are the qualifications and interests of faculty members sufficient to teach all courses, plan, modify and update courses and curricula?	5				
Do the faculty members possess a level of competence that would be obtained through graduate work in the discipline?		4			
Do the majority of faculty members hold Ph.D. degree in their discipline?					1
Do faculty members dedicate sufficient time to research to remain current in their disciplines?					1
Are there mechanisms in place for faculty development?			3		
Are faculty members motivated and satisfied so as to excel in their professions?			3		
Total Encircled Value (TV)	**18**				
Score 6 (S6) = {TV / (No. of Questions * 5)} * 100 * Weight	**10.29**				

Criterion 7 -Institutional Facilities	**Weight = 0.10**				
	Score				
	5	**4**	**3**	**2**	**1**
Does the institution have the infrastructure to support new trends such as e-learning?				2	
Does the library contain technical collection relevant to the program and is it adequately staffed?					1
Are the class rooms and offices adequately equipped and capable of helping faculty carry out their responsibilities?			3		
Total Encircled Value (TV)	**6**				
Score 7 (S7) = {TV / (No. of Questions * 5)} * 100 * Weight	**4.00**				

Criterion 8 - Institutional Support	**Weight = 0.10**				
	Score				
	5	**4**	**3**	**2**	**1**
Is there sufficient support and finances to attract and retain high quality faculty?				2	
Are there an adequate numbers of high quality graduate students, teaching assistants and Ph.D. students?					1
Total Encircled Value (TV)	**3**				
Score 8 (S8) = {TV / (No. of Questions * 5)} * 100 * Weight	**3.00**				

Overall Assessment Score = S1 + S2 + S3 + S4 + S5 + S6 + S7 + S8 =	**61.00**

Remarks:

Weaknesses:

More spacious labs will be helpful in facilitating the practical works and boosting student learning.

More no of will be classes will be helpful in facilitating spacious seating arrangement.

Library resources should be enhanced so as to facilitate students in learning and to improve their knowledge and understanding.