28.Course Specification of Computer Programming (2)

I.Course Identification and General Information:

.1	Course Title:	Computer Programming (2).				
.2	Course Code & Number:	MT201.				
.3	Credit hours:	C.H				TOTAL Cr. Hrs.
		Th.	Seminar	Pr.	Tu.	
		2	-	2	-	3
.4	Study level/ semester at which this course is offered:	Third Year-First Semester.				
.5	Pre –requisite (if any):	Computer Programming (1).				
.6	Co –requisite (if any):	None.				
.7	Program (s) in which the course is offered:	Mechatronics Engineering Program.				
8.	Language of teaching the course:	English Language.				
9.	Location of teaching the course:	Mechatronics Engineering Department.				
10.	Prepared By:	Asst. Prof. Dr. Sami Al-Maqtari.				
11.	Date of Approval:					

II.Course Description:

This course introduces the Object-Oriented Programming (OOP) concepts, principles, and techniques, including classes, objects, inheritance, and polymorphism. All these concepts are illustrated via a contemporary object-oriented programming language called C++. Upon completion, students should be able to use an Object-Oriented (OO) language to develop rather complex programs. The course includes: reference type, function overloading, function overriding, operator overriding, virtual functions, data members and member functions (methods), constructors & destructors, access scope, new I/O operations and how to override them, templates, and an introduction to STL (Standard Template Library).

III.Course Intended learning outcomes (CILOs) of the course		Referenced PILO
a1.	Characterize the Object-Oriented (OO) analysis, design, and programming concepts and gain understanding of the concepts of classes and objects and the method of modeling with them.	A1, A2, A3, A5, A6
a2.	Describe the three main concepts of OOP: encapsulation, polymorphism, and inheritance, show an appreciation of the	

	significance of templates features, and comprehend and apply advanced data types and structures.	
b1.	Analyze the programs in modular reusable blocks (functions, classes, templates) useful to be used in multiple projects and exploit design patterns in the development of software.	B1, B2
b2.	Create software with team-work in mind and evaluate the effectiveness of the OOP development methodology.	
c1.	Demonstrate a working capability of developing Object-Oriented programs efficiently and practice developing computer programs for solving engineering problems.	C1, C2, C5
c2.	Employ the main Object-Oriented Programming concepts (encapsulation, inheritance, and polymorphism) when writing programs and benefit from the reusability of the modular blocks (functions, classes, templates) that were built to develop bigger projects.	
d1.	Review learned methodology to solve advanced problems and cooperate with other developers who write some reusable parts which can be used in bigger projects.	D3, D5, D7
d2.	Assess to OO analysis, design, and programming ideas and evaluate OO solutions and formulate problems as steps so as to be solved systematically.	

(A) Alignment Course Intended Learning Outcomes of Knowledge and Understanding to Teaching Strategies and Assessment Strategies:

Course Intended Learning Outcomes		Teaching Strategies	Assessment Strategies
Characterize the Object-Oriented (OO) analysis, design, and programming concepts and gain understanding of the concepts of classes and objects and the method of modeling with them.	a1.	• Active Lectures. • Tutorials.	• Written Assessment. • Quizzes.
Describe the three main concepts of OOP: encapsulation, polymorphism, and inheritance, appreciation of the significance of templates features, and comprehend and apply advanced data types and structures.	a2.	• Active Lectures. • Tutorials.	• Written Assessment. • Quizzes.

(B) Alignment Course Intended Learning Outcomes of Intellectual Skills to Teaching Strategies and Assessment Strategies:		
Course Intended Learning Outcomes	Teaching Strategies	Assessment Strategies
b1. Analyze the programs in modular reusable blocks (functions, classes, templates) useful to be used in multiple projects and exploit design patterns in the development of software.	• Active Lectures. • Tutorials.	• Written Assessment • Quizzes.
b2. Create software with team-work in mind and evaluate the effectiveness of the OOP development methodology.	• Active Lectures. • Tutorials.	• Written Assessment • Quizzes.

(C) Alignment Course Intended Learning Outcomes of Professional and Practical Skills to Teaching Strategies and Assessment Strategies:		
Course Intended Learning Outcomes	Teaching Strategies	Assessment Strategies
c1. Demonstrate a working capability of developing Object-Oriented programs efficiently and practice developing computer programs for solving engineering problems.	• Active Lectures. • Tutorials.	• Written Assessment. • Quizzes.
c2. Employ the main Object-Oriented Programming concepts encapsulation, inheritance, and polymorphism) when writing programs and benefit from the reusability of the modular blocks (functions, classes, templates) that were built to develop bigger projects.	• Active Lectures. • Tutorials.	• Written Assessment. • Quizzes.

(D) Alignment Course Intended Learning Outcomes of Transferable Skills to Teaching Strategies and Assessment Strategies:		
Course Intended Learning Outcomes	Teaching Strategies	Assessment Strategies
Review learned methodology to solve advanced problems and cooperate with other developers who write some reusable parts which can be used in projects. bigger **d1.**	• Active Lectures. • Tutorials.	• Written Assessment. • Quizzes.
Assess to OO analysis, design, and ideas and evaluate OO programming steps solutions and formulate problems as so as to be solved systematically. **d2.**	• Active Lectures. • Tutorials.	• Written Assessment. • Quizzes.

IV.Course Content:

A – Theoretical Aspect:

Order	Units/Topics List	Learning Outcomes	Sub Topics List	Number of Weeks	Contact Hours
1.	Introduction to C++.	a1, c1, c2, d1.	- C++ as a better C. - Simple computer programs in C++. - Basic input/output statements. - Using primitive data types.	1	2
2.	Introduction to Classes and Objects.	a1, a2, b1, b2, c1, c2, d1, d2.	- The concept of classes, objects, member functions (methods), and data members. - Defining a class and creating an object. - Defining methods (class's behaviors). - Declaring data members (class's attributes). - Calling methods of an object. - Differences between data members and local variables of a function. - Constructors for initializing an object.	2	4

			- Separating class's interface from its implementation.		
3.	Revision over Control Statements.	a1, a2, b1, b2, c1, c2, d1, d2.	- The if selection statement and the if…else selection statements. - The while repetition statement. - Structured programming. - The increment, decrement and assignment operators. The essentials of counter-controlled repetition. - The for and do…while repetition statements. - The switch selection statement. - The break and continue statements. - The logical operators to form complex conditional expressions in control statements. - To avoid the consequences of confusing the equality and assignment operators.	2	4
4.	Classes in More Details.	a1, a2, b1, b2, c1, c2, d1, d2.	- Preprocessor wrappers to avoid multiple definition errors. - Class scope and class members access. - Constructors with default arguments. - Destructors for a proper object termination. - Constant objects and constant methods. - Hierarchical object composition. - friend functions and friend classes. - this pointer for object self-referencing. - Creating and destroying objects dynamically (new & delete keywords). - Static data members and methods.	2	4

5.	Mid-Term Exam.	a1, a2, b1, b2, c1, c2.	- The First 4 Chapters.	1	2
6.	Operator Overloading.	a1, a2, b1, b2, c1, c2, d1, d2.	- The concept of operator overloading and its benefits. - Operator overloading mechanism. - The differences between overloading unary and binary operators. - Casting (converting) objects using operator overloading. - Explicit single-argument constructors to prevent implicit conversion.	1	2
7.	OOP Concepts: Inheritance.	a1, a2, b1, b2, c1, c2, d1, d2.	- Creating classes by inheriting from existing classes. - Understanding the concept of code reusability using inheritance. - The notions of base classes and derived classes. - The protected member access specifier. - Constructors and destructors in inheritance hierarchies. - The calling order of constructors and destructors in inheritance hierarchies. - The different types of inheritance (public, protected, and private)	1	2
8.	OOP Concepts: Polymorphism.	a1, a2, b1, b2, c1, c2, d1, d2.	- The concept of polymorphism. - Virtual functions. - Abstract vs. concrete classes. - Declaring pure virtual functions to create abstract classes. - Safety in using virtual destructors.	1	2
9.	Templates.	a1, a2, b1, b2, c1, c2, d1, d2.	- The concept of templates. - Function templates. - Functions template and function template specializations.	1	2

			- Class templates. - Class templates and class template specializations. - Overloading function template. - The relationships among templates, friends, inheritance, and static members.		
10.	Stream Input/Output.	a1, a2, b1, b2, c1, c2, d1, d2.	- C++ OO input/output streams. - Input and output formatting. - The stream-I/O class hierarchy. - Stream manipulators. - Justification and padding control. - Tying output stream to input stream.	1	2
11.	Data Structures.	a1, a2, b1, b2, c1, c2, d1, d2.	- Self-referential classes and recursion. - Dynamic Memory Allocation & Data Structures. - Linked lists. - Stacks. - Queues. - Trees.	1	2
12.	Standard Template Library (STL).	a1, a2, b1, b2, c1, c2, d1, d2.	- Introduction to containers, iterators, and algorithms. - Sequence containers. - Associative containers. - Container adapters. - Algorithms.	1	2
13.	Final Exam.	a1, a2, b1, b2, c1, c2.	- All the Chapters.	1	2
Number of Weeks /and Units Per Semester				**16**	**32**

B - Practical Aspect:

Order	Tasks/ Experiments	Number of Weeks	Contact Hours	Learning Outcomes
1.	Introducing C++ language compilers and some Integrated Development Environments (IDE)	2	4	a1, b2, c1, d1, d2

2.	Writing simple programs using OO features and discovering the new input/output functions in C++	2	4	a1, a2, b1, b2, c1, c2, d1, d2
3.	Developing C++ programs using classes	2	4	a1, a2, b1, b2, c1, c2, d1, d2
4.	Explore function and operator overloading	1	2	a1, a2, b1, b2, c1, c2, d1, d2
5.	Demonstrating inheritance & polymorphism concepts in C++	2	4	a1, a2, b1, b2, c1, c2, d1, d2
6.	Developing C++ programs using templates	1	2	a1, a2, b1, b2, c1, c2, d1, d2
7.	Overriding the default input/output stream functionalities	1	2	a1, a2, b1, b2, c1, c2, d1, d2
8.	Deploying data structures in C++ programs	1	2	a1, a2, b1, b2, c1, c2, d1, d2
9.	Using Standard Template Library (STL)	2	4	a1, a2, b1, b2, c1, c2, d1, d2
Number of Weeks /and Units Per Semester		**14**	**28**	

V.Teaching strategies of the course:

- Lectures,
- Laboratory,
- Demonstrations,
- Quizzes,
- Practical Classes,
- Assignments.

VI.Assignments:

No	Assignments	Aligned CILOs(symbols)	Week Due	Mark
1.	Participation .	a1, a2, b1, b2, c1, c2, d1, d2	1-14	7.5
Total				**7.5**

Head of the Department
Assoc. Prof. Dr. Abdul-Malik Momin

Quality Assurance Unit
Assoc. Prof. Dr. Mohammad Algorafi

Dean of the Faculty
Prof. Dr. Mohammed AL-Bukhaiti

Academic Development Center & Quality Assurance
Assoc. Prof. Dr. Huda Al-Emad

Rector of Sana'a University
Prof. Dr. Al-Qassim Mohammed Abbas

VII.Schedule of Assessment Tasks for Students During the Semester:

No.	Assessment Method	Week Due	Mark	Proportion of Final Assessment	Aligned Course Learning Outcomes
1.	Participation.	Weekly	7.5	5%	a1, a2, b1, b2, c1, c2, d1, d2
2.	Quizzes.	Once a month	7.5	5%	a1, a2, b1, b2, c1, c2
3.	Mid-Term Exam.	8th week	22.5	15%	a1, a2, b1, b2, c1, c2
4.	Final Exam (Theoretical).	16th week	90	60%	a1, a2, b1, b2, c1, c2
5.	Final Exam (Practical).	13th week	22.5	15%	a1, a2, b1, b2, c1, c2
Total			**150**	**100%**	

VIII.Learning Resources:

- Written in the following order: (Author - Year of publication – Title – Edition – Place of publication – Publisher).

1- Required Textbook(s) (maximum two).

1- P. J. Deitel, H. M., Deitel - C++ How to Program, 8th edition - Pearson Education, Inc. - (ISBN: 9780132662369)

2- Essential References.

1- Gary Bronson – 2011 - A First Book of C++, 4th edition - Cengage Learning
2- Bjarne Stroustrup, The C++ Programming Language, 4th edition - (ISBN: 978-03215638

3- Electronic Materials and Web Sites *etc*.

1- C Programming and C++ Programming: http://www.cprogramming.com/
2- Optimizing C and C++ (http://www.eventhelix.com/realtimemantra/basics/optimizingcandcppcode.htm

IX.Course Policies:

1.	Class Attendance: - The students should have more than 75% of attendance according to rules and regulations of the faculty.
2.	Tardy: - The students should respect the timing of attending the lectures. They should attend within 15 minutes from starting of the lecture.
3.	Exam Attendance/Punctuality: - The student should attend the exam on time. The punctuality should be implemented according to rules and regulations of the faculty for mid-term exam and final exam.

4.	Assignments & Projects: - The assignment is given to the students after each chapter; the student has to submit all the assignments for checking on time.
5.	Cheating: - If any cheating occurred during the examination, the student is not allowed to continue and he has to face the examination committee for enquiries.
6.	Plagiarism: - If one student attends the exam on another behalf; he will be dismissed from the faculty according to the policy, rules and regulations of the university.
7.	Other Policies: - All the teaching materials should be kept out the examination hall and mobile phones are not allowed. - Mutual respect should be maintained between the student and his teacher and also among students. Failing in keeping this respect is subject to the policy, rules and regulations of the university.

Reviewed By	Vice Dean for Academic Affairs and Post Graduate Studies: Asst. Prof. Dr. Tarek A. Barakat. President of Quality Assurance Unit: Assoc. Prof. Dr. Mohammed Algorafi. Head of Mechatronics Engineering Department: Assoc. Prof. Dr. Abdul-Malik Momin.
	Deputy Rector for Academic Affairs Assoc. Prof. Dr. Ibrahim AlMutaa. Assoc. Prof. Dr. Ahmed Mujahed. Asst. Prof. Dr. Munaser Alsubari.

Head of the Department Assoc. Prof. Dr. Abdul-Malik Momin	Quality Assurance Unit Assoc. Prof. Dr. Mohammad Algorafi	Dean of the Faculty Prof. Dr. Mohammed AL-Bukhaiti	Academic Development Center & Quality Assurance Assoc. Prof. Dr. Huda Al-Emad	Rector of Sana'a University Prof. Dr. Al-Qassim Mohammed Abbas

Template for Course Plan of Computer Programming (2)

I.Information about Faculty Member Responsible for the Course:							
Name of Faculty Member	Asst. Prof.Dr. Sami Al-Maqtari	Office Hours					
Location& Telephone No.	771010885	SAT	SUN	MON	TUE	WED	THU
E-mail	dr.samiaziz@gmail.com			10-12			

II.Course Identification and General Information:						
1.	Course Title:	Computer Programming (2)				
2.	Course Number & Code:	MT201.				
3.	Credit hours:	C.H				Total Cr.Hrs.
		Th.	Seminar	Pr.	Tu.	
		2	-	2	-	3
4.	Study level/year at which this course is offered:	Third Year-First Semester.				
5.	Pre –requisite (if any):	Computer Programming (1).				
6.	Co –requisite (if any):	None.				
7.	Program (s) in which the course is offered	Mechatronics Engineering Program.				
8.	Language of teaching the course:	English Language.				
9.	System of Study:	Semesters.				
10.	Mode of delivery:	Lectures and Lab.				
11.	Location of teaching the course:	Mechatronics Engineering Department.				

III.Course Description:

This course introduces the Object-Oriented Programming (OOP) concepts, principles, and techniques, including classes, objects, inheritance, and polymorphism. All these concepts are illustrated via a contemporary object-oriented programming language called C++. Upon completion, students should be able to use an Object-Oriented (OO) language to develop rather complex programs. The course includes: reference type, function overloading, function overriding, operator overriding, virtual functions, data members and member functions (methods), constructors & destructors, access scope, new I/O operations and how to override them, templates, and an introduction to STL (Standard Template Library).

IV.Course Intended learning outcomes (CILOs) of the course (		Referenced PILOs
a1.	Characterize the Object-Oriented (OO) analysis, design, and programming concepts and gain understanding of the concepts of classes and objects and the method of modeling with them.	A1, A2, A3, A5, A6
a2.	Describe the three main concepts of OOP: encapsulation, polymorphism, and inheritance, show an appreciation of the significance of templates features, and comprehend and apply advanced data types and structures.	
b1.	Analyze the programs in modular reusable blocks (functions, classes, templates) useful to be used in multiple projects and exploit design patterns in the development of software.	B1, B2
b2.	Create software with team-work in mind and evaluate the effectiveness of the OOP development methodology.	
c1.	Demonstrate a working capability of developing Object-Oriented programs efficiently and practice developing computer programs for solving engineering problems.	C1, C2, C5
c2.	Employ the main Object-Oriented Programming concepts (encapsulation, inheritance, and polymorphism) when writing programs and benefit from the reusability of the modular blocks (functions, classes, templates) that were built to develop bigger projects.	
d1.	Review learned methodology to solve advanced problems and cooperate with other developers who write some reusable parts which can be used in bigger projects.	D3, D5, D7
d2.	Assess to OO analysis, design, and programming ideas and evaluate OO solutions and formulate problems as steps so as to be solved systematically.	

V.Course Content:

A – Theoretical Aspect:

Order	Units/Topics List	Sub Topics List	Number of Weeks	Contact Hours
1.	Introduction to C++.	- C++ as a better C. - Simple computer programs in C++. - Basic input/output statements. - Using primitive data types.	1	2
2.	Introduction to Classes and Objects.	- The concept of classes, objects, member functions (methods), and data members.	2,3	4

Head of the Department
Assoc. Prof. Dr. Abdul-Malik Momin

Quality Assurance Unit
Assoc. Prof. Dr. Mohammad Algorafi

Dean of the Faculty
Prof. Dr. Mohammed AL-Bukhaiti

Academic Development Center & Quality Assurance
Assoc. Prof. Dr. Huda Al-Emad

Rector of Sana'a University
Prof. Dr. Al-Qassim Mohammed Abbas

		- Defining a class and creating an object. - Defining methods (class's behaviors). - Declaring data members (class's attributes). - Calling methods of an object. - Differences between data members and local variables of a function. - Constructors for initializing an object. - Separating class's interface from its implementation.		
3.	Revision over Control Statements.	- The if selection statement and the if…else selection statements. - The while repetition statement. - Structured programming. - The increment, decrement and assignment operators. The essentials of counter-controlled repetition. - The for and do…while repetition statements. - The switch selection statement. - The break and continue statements. - The logical operators to form complex conditional expressions in control statements. - To avoid the consequences of confusing the equality and assignment operators.	4,5	4
4.	Classes in More Details.	- Preprocessor wrappers to avoid multiple definition errors. - Class scope and class members access. - Constructors with default arguments. - Destructors for a proper object termination. - Constant objects and constant methods. - Hierarchical object composition. - friend functions and friend classes.	6,7	4

		- this pointer for object self-referencing. - Creating and destroying objects dynamically (new & delete keywords). - Static data members and methods.		
5.	Mid-Term Exam.	- The First 4 Chapters.	8	2
6.	Operator Overloading.	- The concept of operator overloading and its benefits. - Operator overloading mechanism. - The differences between overloading unary and binary operators. - Casting (converting) objects using operator overloading. - Explicit single-argument constructors to prevent implicit conversion.	9	2
7.	OOP Concepts: Inheritance.	- Creating classes by inheriting from existing classes. - Understanding the concept of code reusability using inheritance. - The notions of base classes and derived classes. - The protected member access specifier. - Constructors and destructors in inheritance hierarchies. - The calling order of constructors and destructors in inheritance hierarchies. - The different types of inheritance (public, protected, and private)	10	2
8.	OOP Concepts: Polymorphism.	- The concept of polymorphism. - Virtual functions. - Abstract vs. concrete classes. - Declaring pure virtual functions to create abstract classes. - Safety in using virtual destructors.	11	2
9.	Templates.	- The concept of templates. - Function templates.	12	2

		- Functions template and function template specializations. - Class templates. - Class templates and class template specializations. - Overloading function template. - The relationships among templates, friends, inheritance, and static members.		
10.	Stream Input/Output.	- C++ OO input/output streams. - Input and output formatting. - The stream-I/O class hierarchy. - Stream manipulators. - Justification and padding control. - Tying output stream to input stream.	13	2
11.	Data Structures.	- Self-referential classes and recursion. - Dynamic Memory Allocation & Data Structures. - Linked lists. - Stacks. - Queues. - Trees.	14	2
12.	Standard Template Library (STL).	- Introduction to containers, iterators, and algorithms. - Sequence containers. - Associative containers. - Container adapters. - Algorithms.	15	2
13.	Final Exam	- All the Chapters.	16	2
Number of Weeks /and Units Per Semester			**16**	**32**

B - Practical Aspect:

Order	Tasks/ Experiments	Number of Weeks	Contact Hours	Learning Outcomes
1.	Introducing C++ language compilers and some Integrated Development Environments (IDE)	1,2	4	a1, b2, c1, d1, d2

2.	Writing simple programs using OO features and discovering the new input/output functions in C++	3,4	4	a1, a2, b1, b2, c1, c2, d1, d2
3.	Developing C++ programs using classes	5	2	a1, a2, b1, b2, c1, c2, d1, d2
4.	Explore function and operator overloading	6	2	a1, a2, b1, b2, c1, c2, d1, d2
5.	Demonstrating inheritance & polymorphism concepts in C++	7,8	4	a1, a2, b1, b2, c1, c2, d1, d2
6.	Developing C++ programs using templates	9	2	a1, a2, b1, b2, c1, c2, d1, d2
7.	Overriding the default input/output stream functionalities	10	2	a1, a2, b1, b2, c1, c2, d1, d2
8.	Deploying data structures in C++ programs	11,12	2	a1, a2, b1, b2, c1, c2, d1, d2
9.	Using Standard Template Library (STL)	13,14	2	a1, a2, b1, b2, c1, c2, d1, d2
Number of Weeks /and Units Per Semester		**14**	**28**	

VI.Teaching strategies of the course:

- Lectures,
- Laboratory,
- Demonstrations,
- Quizzes,
- Practical Classes,
- Assignments.

VII.Assignments:

No	Assignments	Aligned CILOs(symbols)	Week Due	Mark
1.	Participation.	a1, a2, b1, b2, c1, c2, d1, d2	1-14	7.5
Total				**7.5**

VIII.Schedule of Assessment Tasks for Students During the Semester:

No.	Assessment Method	Week Due	Mark	Proportion of Final Assessment	Aligned Course Learning Outcomes
1.	Participation.	Weekly	7.5	5%	a1, a2, b1, b2, c1, c2, d1, d2
2.	Quizzes.	Once a month	7.5	5%	a1, a2, b1, b2, c1, c2.
3.	Mid-Term Exam.	8th week	22.5	15%	a1, a2, b1, b2, c1, c2.
4.	Final Exam (Theoretical).	16th week	90	60%	a1, a2, b1, b2, c1, c2.
5.	Final Exam (Practical).	13th week	22.5	15%	a1, a2, b1, b2, c1, c2.
Total			**150**	**100%**	

IX.Learning Resources:

• Written in the following order: (Author – Year of publication – Title – Edition – Place of publication – Publisher).

1- Required Textbook(s) (maximum two).

1. P. J. Deitel, H. M., Deitel - C++ How to Program, 8th edition - Pearson Education, Inc. - (ISBN: 9780132662369)

2- Essential References.

1. Gary Bronson – 2011 - A First Book of C++, 4th edition - Cengage Learning
2. Bjarne Stroustrup, The C++ Programming Language, 4th edition - (ISBN: 978-0321563842)

3- Electronic Materials and Web Sites *etc*.

1. C Programming and C++ Programming: http://www.cprogramming.com/
2. Optimizing C and C++ Code: http://www.eventhelix.com/realtimemantra/basics/optimizingcandcppcode.htm

X.Course Policies:

Unless otherwise stated, the normal course administration policies and rules of the Faculty of Engineering apply. For the policy, see: ----------

1.	Class Attendance: - The students should have more than 75% of attendance according to rules and regulations of the faculty.
2.	Tardy: - The students should respect the timing of attending the lectures. They should attend within 15 minutes from starting of the lecture.
3.	Exam Attendance/Punctuality:

	- The student should attend the exam on time. The punctuality should be implemented according to rules and regulations of the faculty for mid-term exam and final exam.
4.	Assignments & Projects: - The assignment is given to the students after each chapter; the student has to submit all the assignments for checking on time.
5.	Cheating: - If any cheating occurred during the examination, the student is not allowed to continue and he has to face the examination committee for enquiries .
6.	Plagiarism: - If one student attends the exam on another behalf; he will be dismissed from the faculty according to the policy, rules and regulations of the university.
7.	Other Policies: - All the teaching materials should be kept out the examination hall and mobile phones are not allowed. - Mutual respect should be maintained between the student and his teacher and also among students. Failing in keeping this respect is subject to the policy, rules and regulations of the university.

Head of the Department Assoc. Prof. Dr. Abdul-Malik Momin	Quality Assurance Unit Assoc. Prof. Dr. Mohammad Algorafi	Dean of the Faculty Prof. Dr. Mohammed AL-Bukhaiti	Academic Development Center & Quality Assurance Assoc. Prof. Dr. Huda Al-Emad	Rector of Sana'a University Prof. Dr. Al-Qassim Mohammed Abbas