Ministry of Higher Education and Scientific Research

Supervision and Scientific Evaluation Body

Quality Assurance and Academic Accreditation Office

Course Description Sample

Subject: -- Object-Oriented Programming ---------

This course description provides a brief survey of the most important characteristics, expected learning output, showing whether students have made full use f the learning opportunities. These characteristics have to be matched with the description of the program.

1. Educational Institution	Shatt Al-Arab University College
2. Department / Center	Computer Science
3. Course Title /Code	
4. Lecturer Name	Hussein Mazin Mohammed
5. Type of Teaching	Attendance
6. Academic Year /Term	Term
7. Total No. of Teaching Hours	120 hours / every week 4 hours
8. Date f Preparing this Course Description	9/29/2022

9. **Course Objectives**

a. Providing students with the most important principles and basics of **Object-Oriented Programming**.
b. The programmer's ability to **analyze**, **design** and implement software solutions to practical problems.
c.Teaching students how to apply the concepts of **inheritance** in the programs that it builds.

d. Providing graduates with the necessary knowledge on **programming** job in organizations.
e. Develop generic programs that do not depend on a specific type of data as well as deal with commonly used generic algorithms and data structures.
f. Providing graduates with the skills of education and creative learning.

10. **Course Output, Methodology and Evaluation**

(A) **Cognitive Objectives**

a. Providing students with the most important principles and basics of **Object-Oriented Programming**.
b. Acquainting students with how to promote their personal knowledge.
c. The ability to analyze, design and implement software solutions to applied problems according to the concepts of object-oriented programming.
d. Enabling students to sharpen their skills in the dynamic work environment.
e. Enabling students to invest their scientific abilities in their working place in the scope of **Programming**.
f. Helping students to get the necessary knowledge to solve problems complex and broken down into subprograms.
g. The ability to apply the concepts of inheritance in programs to achieve the largest possible reduction in code.
h. Discovering **programming errors** that the student faces.
i. Use of generic software that does not depend on a specific type of data as well as dealing with commonly used generic algorithms and data structures.

(B) **Skill Objectives Related to the Program**:

a. Identify programming problems and how to solve them.
b. Real analysis of the problem and its translation into practical issues.
c. Visualize the solution of the problem and break it down into sub-problems using classes.
d. A mental visualization of code, objects, and classes, and how they can be used in future projects and schemes.

Methods of Teaching and Learning

a. Using already- prepared lectures.
b. Using up-to-date data shows.
c. - Using various means to increase understanding and clarification through program analysis and to demonstrate this in practice in the lab.
d. Homework
e. Adopting group discussions.

Methods of Evaluation

a. Oral tests
b. Monthly tests
c. Daily quizzes
d. Students' Regular Attendance
e. Practical exams (lab)

(C) **Sentimental and Value Objectives**

a. Realizing ethical objectives.
b. Commitment to university traditions.
c. Compliance with the University Instructions and the Ministry Regulations.
d. Promoting students' personal abilities in educational scopes and how to behave well with others.
e. Conduct meaningful discussions with international software and engineering teams to improve software and engineering skills.
f. Active participation in the service of his peers and the service of the community and the country.

Methods of Teaching and Learning

a. Lectures on university instructions.
b. Educational guidance lectures.
c. Continuous directing.
d. Visiting State and private institutions.
e. Showing practical cases.
f. Group discussions.

Methods of Evaluation

a. Daily quizzes.
b. Classroom discussions and commitment to ethics and sublime values.
c. Special marks for class activities.
d. Monthly and quarterly evaluation.
e. The homework assigned to the student of writing and implementing the required programs.
f. Reports and studies on the methods of writing programs and methods of programming thinking.

D) **General and Qualitative Skills (other skills related to the ability of employment and personal development)**

a. Enabling students to acquire the skill communicate with others and understand and read team programs.
b. Enabling students to apply creative thinking in programming.
c. Analysis and investigation from analyzing the problem written in the form of sub-problems in order to write a program that represents the real problem.
d. Enabling students to use modern methods of analysis and conclusions.
e. Enabling students to use of modern technology.

11. **Course Structure**

Week	No of Hours	Required Learning Output	Title of Subject	Teaching Method	Evaluation

1-2	4 theoretical lectures + 4 lab	The student should be able to recall the basic concepts and tools of Structured Programming using C++ language	C++ Review (Program structure, namespace, identifiers, variables, constants, enum, operators, typecastings, control structures and functions).	- Theoretical lectures - Practical lectures (lab)	- Achievement test - Discussions and questions
3	2 theoretical lectures + 2 lab	The student should be able to understand the basic concepts of object-oriented programming	Introduction to Object-Oriented Programming in C++.	- Theoretical lectures - Practical lectures (lab)	- Achievement test - Discussions and questions
4-8	10 theoretical lectures + 10 lab	The student should be able to analyze, design and implement software solutions to applied problems according to the concepts of object-oriented programming	Objects and Classes (Basics of objects an classes in C++, private and public members, static data and function members, constructors and their types, destructors and operator overloading).	- Theoretical lectures - Practical lectures (lab) -Discussions	- Achievement test - Discussions and questions - Homeworks
9-14	12 theoretical lectures + 12 lab	The student should be able to apply the concepts of inheritance in the programs that he builds to achieve the largest possible reduction in the code.	Inheritance (Concepts of Inheritance, types of inheritance: single, multiple, multilevel, hierarchical, hybrid, protected members, overriding, virtual base class).	- Theoretical lectures - Practical lectures (lab) -Discussions	- Achievement test - Discussions and questions - Homeworks
15-19	10 theoretical lectures + 10 lab	The student should be able to understand, design and apply programming problems that include the concept of polymorphism	Polymorphism (Pointers in C++, Pointes and Objects, this pointer, virtual and pure virtual functions, Implementing polymorphism).	- Theoretical lectures - Practical lectures (lab) -Discussions	- Achievement test - Discussions and questions - Homeworks
20-24	10 theoretical lectures + 10 lab	The student should be able to deal with files in their various forms to store and retrieve data	I/O and File management (Concepts of streams, cin and cout objects, C++ stream classes, Unformatted and formatted I/O, manipulators, File stream, C++ File stream	- Theoretical lectures - Practical lectures (lab) -Discussions	- Achievement test - Discussions and questions - Homeworks

			classes, File management functions, File modes, Binary and random files).		
25-30	12 theoretical lectures + 12 lab	The student should be able to develop general programs that do not depend on a specific type of data, as well as deal with commonly used general algorithms and data structures, and also be able to design programs that have the ability to deal with error cases that occur during program implementation.	Templates, Exceptions and STL (What is template? function templates and class templates, Introduction to exception, try-catch-throw, multiple catch, catch all, rethrowing user defined exceptions, Overview and use of Standard Template Library).	- Theoretical lectures - Practical lectures (lab) -Discussions	- Achievement test - Discussions and questions - Homeworks

12.**Infrastructure**

a. Textbooks	**"Object-Oriented Programming in C++"**, 4th Edition, Robert Lafore, Sams Publishing, 2002.
b. References	
c. Recommended books and periodicals (journals, reports, etc.)	**"CPA: Programming Essentials in C++"**, C++ INSTITUTE, 2016.
d. Electronic references, internet websites, etc	**"C++ Tutorial"**, tutorialspoint. https://www.tutorialspoint.com/cplusplus/index.htm

13. **The Plan of Improving the Course**

a. Increasing the number of theoretical lecture hours to three hours instead of two hours, where the additional hour is devoted to discussing additional programming methods and examples, with the expansion of holding discussions, which leads to a better understanding of the course subjects.
b. Be informed of the development and interest of other countries in the field of programming.

c. Be informed of research work published in national and international journals in the field of programming.