جامعة فيلادلفيا Philadelphia University	اسم النموذج : خطة المادة الدراسية	رمز النموذج : QFO-AP-VA-008
	الجهة المصدرة: نائب الرئيس للشؤون الأكاديمية	رقم الإصدار: (Rev) 2
	الجهة المدققة : اللجنة العليا لضمان الجودة	تاريخ الإصدار: 2021-5-4
		عدد صفحات النموذج : 4

Course Title: **Programming Fundamentals (1)**			**Course code:** 0750113		
Course Level: 1			**Course prerequisite (s) and/or corequisite(s):**		
Lecture Time:			**Credit hours: 3**		
UR ☐	**FR** ■	**DR** ☐	**C** ☐	**E** ☐	

Academic Staff Specifics

Name	Rank	Office No. and Location	Office Hours	E-mail Address
Enas AbuSamra	Instructor	IT Office:313	11:00 – 12:30 Sat, Mon 9:30 – 11:00 Sun, Tue	eabusamra@philadelphia.edu.jo

The Learning Style Used in Teaching the Course

The Learning Style			
Blended Learning ☐			
Electronic Learning ☐			
Face-to-Face Learning ☐			
Face-to-Face	Electronic	Blended	Percentage
%100			

Course/Module Description:
This module focuses on problem solving strategies and the use of algorithmic language to describe such problem solving. It introduces the principles of procedural programming, data types, control structures, data structures and functions, data representation on the machine level.

Course/Module Objectives:
This module aims to introduce computer programming and emphasis in problem solving on the fundamentals of structured design using the principles of Top-Down problem-solving strategy (divide and conquer). This includes development, testing, implementation, documentation.
The module also aims to explore the logic of programming via the algorithm concepts and implement them in programming structures including functions, arrays, and pointers.

Course/ module components

- **Textbook:**
  - D.S. Malik , Thomson, C++ Programming: From Problem Analysis to Program Design, 8th Edition, Course Technology, 2018.

- **Supporting material(s):** Lectures handouts
  Introduction to Computer Science and Programming (Spring 2011) (MIT)
  Introduction to C++ (MIT)

Teaching methods:
Duration: 16 weeks, 80 hours in total
Lectures: 32 hours (2 hours per week),
Tutorials: 16 hours (1 per week),
Laboratories: 32 hours, 2 per week

Learning outcomes

A- Knowledge and understanding

A1. Identify a wide range of principles and tools available to the software developer, such as design methodologies, choice of algorithm, language, software libraries and user interface technique.

B- ***Intellectual skills (thinking and analysis).***

B1. Analyze a wide range of problems and provide solutions through suitable algorithms, structures, diagrams, and other appropriate methods

C- ***Practical skills***

C1. Assess the balance between self-reliance and seeking help, when necessary, in new situations.

D- ***Transferable Skills***

D1. Prepare and deliver coherent and structured verbal and written technical reports.
D2. Design, write, and debug computer programs in appropriate languages.

Learning outcomes achievement

- ✓ Development: A1 are developed through the lectures and laboratory sessions, C1, D1, D2 are developed through Tutorials and Lab sessions, B1, D1, and D2 are developed through Homework.

- ✓ Assessment: A1, B1, D1, and D2 and are assessed through Quizzes, written exams, and Practical Works Exams. D1, D2 and C2 are assessed through Homework Exam.

Assessment instruments

Allocation of Marks	
Assessment Instruments	**Mark**
Mid Exam	**30%**
Final examination	**40%**
Lab works, Quizzes, and tutorial contributions	**30%**
Total	**100%**

Course/Module Academic Calendar

Week	Basic and support material to be covered	Homework/reports and their due dates
(1)	***Problem Solving***: process, Analyze (requirement, Design algorithm, Tracing algorithm, Example, Design problems).	**Lab work #1**
(2)	***Problem Analysis***: Algorithm discovery, Algorithm design strategies**,** Stepwise refinement**,** Control requirements, Implementing algorithm, Conclusion.	**Lab work #2**
(3)	***Data Definition Structures***: Types, constants, variables, Expressions: Arithmetic, Logical; Precedence rules.	**Lab work #3**
(4)	***Control Structures***: Sequencing; Input and output statements; Assignment statement.	**Lab work #4**
(5)	***Control Structures***: Selection: one-way (if .. then), two-way (if .. then .. else), multiple (switch).	**Lab work #5**
(6)	***Control Structures***: Repetition (counter-controlled loop).	**Lab work #6**
(7) Mid Exam	***Control Structures***: Repetition (Conditional Loop).	**Lab work #7**
(8)	***Control Structures***: Nested Loops, Break and Continue.	**Lab work #8**
(9)	***Control Structures***: Combination.	**Lab work #9**
(10)	***Functions:*** Parameters definition and passing (functions depth look); prototypes	**Lab work #10**
(11)	***Functions:*** Parameters definition and passing (Scope: local and global variables), static variables.	**Lab work #11**
(12)	Application of Functions.	**Lab work #12**
(13)	***Data Structures:*** One dimensional arrays.	**Lab work #13**
	Data Structures: Two dimensional arrays.	**Lab work #14**

(14)		
(15)	***Data Structures:*** Combination (Array + Functions) .	**Lab work #15**
(16) Final Examination	Review and final Exam	**Lab work #16**

Expected workload:
On average students need to spend 3 hours of study and preparation for each 50-minute lecture/tutorial.

Attendance policy:
Absence from lectures and/or tutorials shall not exceed 15%. Students who exceed the 15% limit without a medical or emergency excuse acceptable to and approved by the Dean of the relevant college/faculty shall not be allowed to take the final examination and shall receive a mark of zero for the course. If the excuse is approved by the Dean, the student shall be considered to have withdrawn from the course.

Module references
Students will be expected to give the same attention to these references as given to the Module textbook(s)

1. P. Deitel & H. Deitel, **C++ How to program,** Pearson Education Limited, 2013.
2. Malik, D. S., C++ Programming: Program Design including Data Structures, MA Course Technology, 2009
3. Friedman Frank and Koffman Elliot B., "*Problem Solving, Abstraction and Design using C++"*, Pearson Education , 2011.
4. Lambert Kenneth and Nance Douglas W., "*Understanding Programming and Problem Solving With C++"*, PWS Publishing Company, Fourth Edition. 1996
5. Forouzan, B. A. & R. F. Gilberg. *"Computer Science: A Structured Programming Approach using C"*, Second Edition, Pacific Grove, CA: Brooks/Cole, 2001
6. Bruce Eckel, *"Thinking in C++"*, Second Edition, Prentice Hall, 2000.
7. Herbert Schildt, *"Teach Yourself C++"*, Third Edition, McGraw-Hill. 1998.
8. Lospinoso, J., C++ Crash Course: A Fast-Paced Introduction, No Starch Press; Illustrated Edition, 2019
9. Code Quickly, Learn C++ Quickly: A Complete Beginner's Guide to Learning C++, Even If You're New to Programming (Crash Course With Hands-On Project) , Drip Digital, 2020

Website(s):

- www.cee.hw.zc.uk/~pjbk/pathways/cpp1/cpp1.html
- www.edm2.com/0507/introcpp1.html
- www.doc.ic.ac.uk/~wjk/C++intro
- www.cprogramming.com/tutorial.html
- www.cs.umd.edu/users/cml/cstyle/ellemtel-rules.html
- www.deakin.edu.au/~agoodman/Ctutorial.html
- www.tldp.org/howto/c++programming.howto.html
- www.vb-bookmark.com/cpptutorial.html