MODULE DESCRIPTION FORM

نموذج وصف المادة الدراسية

| **Module Information**  **معلومات المادة الدراسية** | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Title** | Computer Programming 1 | | | | **Module Delivery** | | |
| **Module Type** | Basic | | | | * **☒ Theory** * **☐ Lecture** * **☒ Lab** * **☐ Tutorial** * **☐ Practical** * **☐ Seminar** | | |
| **Module Code** | ITC320011 | | | |
| **ECTS Credits** | 6.00 | | | |
| **SWL (hr/sem)** | 150 | | | |
| **Module Level** | | 1 | **Semester of Delivery** | | | | 1 |
| **Administering Department** | | BID | **College** | BMIC | | | |
| **Module Leader** | Ahmed Uday | | **e-mail** | [ahmed.oday@uoitc.edu.iq](mailto:ahmed.oday@uoitc.edu.iq) | | | |
| **Module Leader’s Acad. Title** | |  | **Module Leader’s Qualification** | | | |  |
| **Module Tutor** | Name (if available) | | **e-mail** | E-mail | | | |
| **Peer Reviewer Name** | | jwan k. alwan | **e-mail** | jwanism@uoitc.edu.iq | | | |
| **Scientific Committee Approval Date** | | 18/06/2023 | **Version Number** | | | 1.0 | |

| **Relation with other Modules**  **العلاقة مع المواد الدراسية الأخرى** | | | |
| --- | --- | --- | --- |
| **Prerequisite module** | None | **Semester** |  |
| **Co-requisites module** | None | **Semester** |  |

| **Module Aims, Learning Outcomes and Indicative Contents**  **أهداف المادة الدراسية ونتائج التعلم والمحتويات الإرشادية** | |
| --- | --- |
| **Module Objectives**  **أهداف المادة الدراسية** | The objectives of a beginner's computer programming course can vary depending on the specific goals and context of the course. However, here are some common objectives:  1. Introduce Programming Concepts: The course should aim to familiarize students with fundamental programming concepts such as variables, data types, control structures, functions, and basic problem-solving techniques.  2. Develop Problem-Solving Skills: The course should help students develop their ability to break down complex problems into smaller, manageable tasks and apply programming techniques to solve them.  3. Provide Hands-On Experience: Students should have ample opportunities to practice programming through hands-on exercises, coding assignments, and projects. This practical experience helps reinforce the concepts learned and build programming skills. |
| **Module Learning Outcomes**  **مخرجات التعلم للمادة الدراسية** | learning outcomes for a beginner's computer programming course:  1. Knowledge of Programming Concepts: Students will gain a solid understanding of fundamental programming concepts such as variables, data types, control structures, functions, and basic algorithms.  2. Proficiency in a Programming Language: Students will become proficient in using a specific programming language, such as Python, Java, or JavaScript, to write and execute basic programs.  3. Problem-Solving Skills: Students will develop the ability to break down complex problems into smaller, manageable tasks and use programming techniques to solve them.  4. Debugging and Troubleshooting: Students will learn how to identify and fix common errors and bugs in their programs using debugging tools and techniques.  5. Code Design and Organization: Students will be able to design and organize their code effectively, utilizing concepts like modularity, reusability, and good coding practices.  6. Collaboration and Teamwork: Students will have opportunities to collaborate on programming projects, fostering teamwork and enhancing their ability to work in a collaborative programming environment.  7. Documentation and Code Readability: Students will understand the importance of documenting their code and writing clean, readable code that is easily understood by others |
| **Indicative Contents**  **المحتويات الإرشادية** | Indicative content includes the following.  1. Introduction to Programming  - Overview of programming languages and their significance  - Setting up the development environment  - Basics of writing and running a program  2. Variables and Data Types  - Introduction to variables and their role in programming  - Common data types: integers, floats, strings, booleans  - Variable assignment and manipulation  3. Control Structures and Decision Making  - Conditional statements: if, if-else, nested conditions  - Looping structures: for loop, while loop  - Iteration and control flow  4. Functions and Modular Programming  - Understanding functions and their purpose  - Defining and calling functions  - Parameters, return values, and scope |

| **Learning and Teaching Strategies**  **استراتيجيات التعلم والتعليم** | |
| --- | --- |
| **Strategies** | 1. Hands-on Practice: Provide ample opportunities for students to engage in hands-on programming exercises and projects. This practical experience helps solidify the concepts learned and build programming skills.  2. Incremental Learning: Break down complex programming concepts into smaller, more manageable topics. Introduce concepts gradually, building upon previously learned material. This approach helps students grasp new concepts without feeling overwhelmed.  3. Code Examples and Demonstrations: Use code examples and demonstrations to illustrate programming concepts and best practices. Show step-by-step examples of how to solve problems and explain the reasoning behind each step. |

| **Student Workload (SWL)**  **الحمل الدراسي للطالب محسوب لـ ١٥ اسبوعا** | | | |
| --- | --- | --- | --- |
| **Structured SWL (h/sem)**  **الحمل الدراسي المنتظم للطالب خلال الفصل** | 63 | **Structured SWL (h/w)**  **الحمل الدراسي المنتظم للطالب أسبوعيا** | 4 |
| **Unstructured SWL (h/sem)**  **الحمل الدراسي غير المنتظم للطالب خلال الفصل** | 87 | **Unstructured SWL (h/w)**  **الحمل الدراسي غير المنتظم للطالب أسبوعيا** | 6 |
| **Total SWL (h/sem)**  **الحمل الدراسي الكلي للطالب خلال الفصل** | **150** | | |

| **Module Evaluation**  **تقييم المادة الدراسية** | | | | | |
| --- | --- | --- | --- | --- | --- |
| **As** | | **Time/Number** | **Weight (Marks)** | **Week Due** | **Relevant Learning Outcome** |
| **Formative assessment** | **Quizzes** | 2 | 10% (10) | 6, 12 | LO #1, #4 and #6, |
| **Assignments** | 2 | 10% (10) | 13 | LO #3, #4 |
| **Projects / Lab.** | 1 | 10% (10) | Continuous | All |
| **Report** | 1 | 10% (10) | 14 | LO 5, and 7 |
| **Summative assessment** | **Midterm Exam** | 2hr | 10% (10) | 7 | LO #1 - #7 |
| **Final Exam** | 3hr | 50% (50) | 16 | All |
| **Total assessment** | | | 100% (100 Marks) |  |  |

| **Delivery Plan (Weekly Syllabus)**  **المنهاج الاسبوعي النظري** | |
| --- | --- |
| **Week** | **Material Covered** |
| **Week 1** | Overview of programming languages and their importance |
| **Week 2** | Writing and running a simple "Hello, World!" program and Basic programming syntax and conventions |
| **Week 3** | Variables and Data Types |
| **Week 4** | Conditional statements (if, if-else, nested if statements) |
| **Week 5** | Looping structures (for, while loops) |
| **Week 6** | Controlling program flow with branching and decision-making statements |
| **Week 7** | Mid-Fist Exam |
| **Week 8** | Functions 1 Programming |
| **Week 9** | Functions 2 Programming |
| **Week 10** | Modular Programming |
| **Week 11** | File Input/Output |
| **Week 12** | Handling file exceptions and errors |
| **Week 13** | File management and manipulation |
| **Week 14** | Project discusses |
| **Week 15** | Reviwe |

| **Delivery Plan (Weekly Lab. Syllabus)**  **المنهاج الاسبوعي للمختبر** | |
| --- | --- |
| **Week** | **Material Covered** |
| **Week 1-2** | Writing and running a simple "Hello, World!" program and Basic programming syntax and conventions |
| **Week 3-4** | Variables and Data Types |
| **Week 5-6** | Conditional statements (if, if-else, nested if statements) |
| **Week 7-8** | Looping structures |
| **Week 9-10** | Functions Programming |
| **Week 11-12** | Modular Programming |
| **Week 13-14** | File Input/Output |
| **Week 15** | Presentations and Discussions |

| **Learning and Teaching Resources**  **مصادر التعلم والتدريس** | | |
| --- | --- | --- |
|  | **Text** | **Available in the Library?** |
| **Required Texts** | "Python Crash Course" by Eric Matthes: A beginner-friendly book for learning Python. | Yes |
| **Recommended Texts** | "Automate the Boring Stuff with Python" by Al Sweigart: This book focuses on practical Python programming, teaching you how to automate repetitive tasks and work with files, spreadsheets, web scraping, and more. | No |

| **Grading Scheme**  **مخطط الدرجات** | | | | |
| --- | --- | --- | --- | --- |
| **Group** | **Grade** | **التقدير** | **Marks %** | **Definition** |
| **Success Group**  **(50 - 100)** | **A -** Excellent | **امتياز** | 90 - 100 | Outstanding Performance |
| **B -** Very Good | **جيد جدا** | 80 - 89 | Above average with some errors |
| **C -** Good | **جيد** | 70 - 79 | Sound work with notable errors |
| **D -** Satisfactory | **متوسط** | 60 - 69 | Fair but with major shortcomings |
| **E -** Sufficient | **مقبول** | 50 - 59 | Work meets minimum criteria |
| **Fail Group**  **(0 – 49)** | **FX –** Fail | **راسب (قيد المعالجة)** | (45-49) | More work required but credit awarded |
| **F –** Fail | **راسب** | (0-44) | Considerable amount of work required |
|  |  |  |  |  |
| **Note:** Marks Decimal places above or below 0.5 will be rounded to the higher or lower full mark (for example a mark of 54.5 will be rounded to 55, whereas a mark of 54.4 will be rounded to 54. The University has a policy NOT to condone "near-pass fails" so the only adjustment to marks awarded by the original marker(s) will be the automatic rounding outlined above. | | | | |