MODULE DESCRIPTION FORM

نموذج وصف المادة الدراسية

| **Module Information**  **معلومات المادة الدراسية** | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Module Title** | Computer Programming II | | | | **Module Delivery** | | |
| **Module Type** | Basic | | | | * **☒ Theory** * **☐ Lecture** * **☒ Lab** * **☐ Tutorial** * **☐ Practical** * **☐ Seminar** | | |
| **Module Code** | ITC320012 | | | |
| **ECTS Credits** | 6.00 | | | |
| **SWL (hr/sem)** | 150 | | | |
| **Module Level** | | 1 | **Semester of Delivery** | | | | 2 |
| **Administering Department** | | BID | **College** | BMIC | | | |
| **Module Leader** | Ahmed Oday | | **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** | Computer Programming 1 / BMI112 | **Semester** | 1 |
| **Co-requisites module** | None | **Semester** |  |

| **Module Aims, Learning Outcomes and Indicative Contents**  **أهداف المادة الدراسية ونتائج التعلم والمحتويات الإرشادية** | |
| --- | --- |
| Module Objectives  أهداف المادة الدراسية | Module objectives for a Python language programming course can vary depending on the level and focus of the course, as well as the intended audience. However, here are some common objectives that instructors might include:   * Understanding Python Basics:   + Learn the syntax and semantics of the Python programming language.   + Understand basic data types such as integers, floats, strings, lists, tuples, dictionaries, etc.   + Understand variables, expressions, and basic operations. * Control Flow:   + Learn how to use control structures such as if statements, loops (for and while), and conditional expressions.   + Understand how to use branching and looping to control the flow of a program. * Functions and Modules:   + Understand how to define and call functions.   + Learn about function parameters and return values.   + Understand the concept of modules and how to import and use them.   + Introduction to standard library modules and external packages. |
| Module Learning Outcomes  مخرجات التعلم للمادة الدراسية | Module Learning Outcomes for a Computer programming II course typically include a range of skills and competencies that students are expected to achieve by the end of the module:   1. Basic Syntax and Data Structures: Understand and apply fundamental Python syntax rules, including variables, data types (integers, floats, strings, lists, tuples, dictionaries), and basic operators. 2. Control Flow: Demonstrate proficiency in control flow structures such as loops (for, while), conditional statements (if, elif, else), and exception handling (try, except, finally). 3. Functions and Modules: Define and utilize functions to organize code into reusable blocks. Import and use modules to extend Python's functionality and encapsulate related code. 4. File Handling: Read from and write to files using Python's file handling mechanisms, including open(), close(), read(), write(), and file modes. |
| Indicative Contents  المحتويات الإرشادية | The indicative contents for a Python language programming course typically cover a range of topics, progressing from basic concepts to more advanced techniques. Below is an outline of indicative contents for such a course:  Overview of Python and its features  Installing Python and setting up the development environment  Running Python code: interactive mode, script mode  Basic syntax and data types: variables, numbers, strings, lists, tuples, dictionaries  Basic operations and built-in functions  Control flow: if statements, loops (for, while)  Functions: defining functions, parameters, return values  Scope and lifetime of variables  Function arguments: positional, keyword, default, variable-length |

| Learning and Teaching Strategies  استراتيجيات التعلم والتعليم | |
| --- | --- |
| Strategies | Designing an effective Python language programming course requires thoughtful consideration of various strategies to engage students, facilitate learning, and achieve desired learning outcomes. Here are some strategies to consider:Hands-on Learning: Provide plenty of opportunities for students to write and run Python code. Hands-on exercises, coding labs, and projects help reinforce learning and build practical skills.Project-Based Learning: Incorporate projects throughout the course that require students to apply Python programming concepts to solve real-world problems. Projects encourage creativity, critical thinking, and collaboration.Peer Learning and Collaboration: Encourage peer learning through group projects, pair programming, and collaborative problem-solving activities. Peer interaction fosters a supportive learning environment and allows students to learn from each other's experiences. |

| **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) | 5 and 10 | LO #1, #2 and #9 |
| **Assignments** | 2 | 10% (10) | 2 and 12 | LO #3, #4 and #8 |
| **Projects / Lab.** | 1 | 10% (10) | Continuous | All |
| **Report** | 1 | 10% (10) | 13 | LO #5, #8 and #9 |
| **Summative assessment** | **Midterm Exam** | 2hr | 10% (10) | 7 | LO #1 - #5 |
| **Final Exam** | 3hr | 50% (50) | 16 | All |
| **Total assessment** | | | 100% (100 Marks) |  |  |

| **Delivery Plan (Weekly Syllabus)**  **المنهاج الاسبوعي النظري** | |
| --- | --- |
| **Week** | **Material Covered** |
| **Week 1** | Functions (defining functions, parameters, return statements) |
| **Week 2** | Tuples and Tuple Manipulation |
| **Week 3** | Introduction to arrays and lists in Python |
| **Week 4** | Sets and Set Operations |
| **Week 5** | Boolean Logic and Truthiness |
| **Week 6** | Dictionaries and Dictionary Manipulation |
| **Week 7** | Mid-First Exam |
| **Week 8** | Introduction to Libraries such as NumPy and Pandas |
| **Week 9** | List Comprehensions |
| **Week 10** | Introduction to Documentation and Docstrings |
| **Week 11** | Introduction to Web Development with Flask or Django |
| **Week 12** | Introduction to GUI Programming with libraries like Tkinter or PyQt I |
| **Week 13** | Introduction to GUI Programming with libraries like Tkinter or PyQt II |
| **Week 14** | Version Control with Git and GitHub |
| **Week 15** | Second Exam |

| **Delivery Plan (Weekly Lab. Syllabus)**  **المنهاج الاسبوعي للمختبر** | |
| --- | --- |
| **Week** | **Material Covered** |
| **Week 1** | Lab 1:Functions (defining functions, parameters, return statements) |
| **Week 2** | Lab 2:Lab 2:Tuples and Tuple Manipulation |
| **Week 3** | Lab 3:Introduction to arrays and lists in Python |
| **Week 4** | Lab 4: Sets and Set Operations |
| **Week 5** | Lab 5: Boolean Logic and Truthiness |
| **Week 6** | Lab 6: Dictionaries and Dictionary Manipulation |
| **Week 7** | Mid-First Exam |
| **Week 8** | Lab 7: Introduction to Libraries such as NumPy and Pandas |
| **Week 9** | Lab 8: List Comprehensions |
| **Week 10** | Lab 9: Introduction to Documentation and Docstrings |
| **Week 11** | Lab 10: Introduction to Web Development with Flask or Django |
| **Week 12** | Lab 11: Introduction to GUI Programming with libraries like Tkinter or PyQt I |
| **Week 13** | Lab 12: Introduction to GUI Programming with libraries like Tkinter or PyQt II |
| **Week 14** | Lab 13: Version Control with Git and GitHub |
| **Week 15** | Second Exam |

| **Learning and Teaching Resources**  **مصادر التعلم والتدريس** | | |
| --- | --- | --- |
|  | **Text** | **Available in the Library?** |
| **Required Texts** | Official Python Documentation: The official Python documentation provides comprehensive guides, tutorials, and references suitable for both beginners and advanced users. You can find it at Python.org. | Yes |
| **Recommended Texts** | Learning Python" by Mark Lutz. | 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. | | | | |