Text

Description automatically generated

**South East Technological University**

**FACULTY OF LIFELONG LEARNING**

**ASSIGNMENT**

**Higher Diploma in Computing KRSIT\_H**

|  |  |  |
| --- | --- | --- |
| **Programme Title & Programme Code** | CW\_KRSIT\_H | |
| **Module Title** | Scripting | |
| **Assignment Number** | One | |
| **Assignment Type** | Continuous Assessment Timed test | |
| **This assignment addresses the following Learning Outcomes:** | LO 1 | Automate everyday interactions with the underlying operating system |
| LO 3 | Exploit scripting within a collection of domain areas. |
| **Weighting** | 20% | |
| **Submission Date** | 02-Dec-2024  Time allotted 2.5 hours | |

**Continuous Assessment Submission Guidelines**

Assignments must be submitted via Blackboard. If assignments are not submitted via Blackboard, this will be regarded as a non-submission.

**Extension Policy**

Undergraduate and postgraduate extensions cannot be granted by your lecturer. Such extensions can only be granted by the Faculty of Lifelong Learning once a completed extension form and supporting documentation is returned online.

Students can apply for extensions at:

<https://www.itcarlow.ie/study/lifelong-learning/lll-forms/extension-request-form.htm>

Extensions must be sought in advance of the submission date. Extensions will not be granted retrospectively.

The circumstances under which an extension request will be considered include, but are not limited to:

* Serious personal/family/business reasons
* Where a student is representing their country or university

An extension request is **unlikely** to be considered under the following circumstances:

* Minor illnesses such as a common cold
* Holidays during the academic year
* Multiple assignments due at the one time
* Failure to plan study schedule
* Debs/weddings/social events
* IT and/or computer failure

The Faculty of Lifelong Learning reserve the right to request supporting documentation. If you are applying for an extension that exceeds **5 days** you must submit supporting documentation (e.g. letter from a doctor, employer, line manager etc.) so that any prolonged absence can be verified.

Please note that loss of/damage to a USB stick is not considered a valid reason for an extension. To avoid any unnecessary distress, please ensure that you **back up your work** regularly as you undertake your assignment. By registering with free online storage services such as Dropbox or Skydrive you can save your work online and access it at any computer. Alternatively use an external hard drive/or just email it to yourself.

Failure to submit a piece of assessment may result in a grade of 0.

**Plagiarism Policy**

Assignments which have copied work from websites, from other authors, from other students or any other sources will receive a grade of 0. All instances of plagiarism must be reported to the Head of Department who, in turn, is obliged to report them to the Registrar. Students who receive this grade may be asked to justify their actions to the University’s plagiarism panel. Students must apply to the examination board at the end of the academic year to re-submit their work in such instances. Please note that copying verbatim from original sources is unacceptable even if you provide references.

# Assignment Details

**Tip for students:** complete the final column with a ‘ü’ for each row to show you have read and understood the detail involved – if not, write in your query and contact your lecturer for clarification.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Module:** | Scripting | | |  |
| **Nature of Assignment:** | Coding – Familiarity with Python structures, datatypes and syntax | | |  |
| **Assignment Weighting:** | 20% of overall module marks | | |  |
| **Feedback - Dates and Nature:** | After submission, initial feedback will be provided before the 8th of Jan 2025. | | |  |
| **Aim of Assignment:** | The aim of this assignment is to assess your knowledge of the Python language and of its usage.  In particular, the assignment will enable you to demonstrate your knowledge of Python data structures including dictionaries, lists and string, and their manipulation. | | |  |
| **Description (include link to additional detailed brief if required, e.g. case study, drawings, etc.):** | In this timed test you will be building a small textual adventure game.  The player will navigate different areas of a castle and will try to avoid or defeat a dragon and claim the throne.  The skeleton for the game will be provided to you, and you will need to build the parts that will be requested. | | |  |
|  |  |  |  |  |

# Submission Details and marked awarded

## Introduction

You are provided with the template of the game; you will have to fill the missing functions. To simplify the work, the template called script\_timed\_test\_one\_template.py is present as part of the material alongside this brief. Anyway, the template is also copied here in the text of this brief.

## Requirements for the submission

The map for the game is provided to you as a dictionary and is illustrated below: the dictionary contains locations, directions, and items. The game is limited, there can only be 0 items or 1 item in a room.

g\_locations = {

'Tower': {'north': 'Throne Room', 'east': 'Armory', 'south': 'Dungeon', 'item': 'map'},

'Throne Room': {'south': 'Tower', 'item': 'crown'},

'Armory': {'west': 'Tower', 'north': 'Library', 'item': 'sword'},

'Library': {'south': 'Armory', 'item': 'spellbook'},

'Dungeon': {'north': 'Tower', 'item': 'dragon'}

}

The structure for what is called the game mainloop, i.e. the part of the game that repeats over and over until the player wins or loses is shown below, while this is given to you, you will have to fill the parts which are missing, and implement the missing functions:

def game\_mainloop(first\_location: str, locations: dict) -> bool:

current\_location = first\_location

inventory = []

show\_game\_options()

while True:

show\_status(current\_location, locations)

move = get\_command()

if move:

action, target = move

if action == "go":

current\_location = move\_to\_next\_location( current\_location, target, locations)

elif action == "get":

pickup\_item(target, inventory, current\_location, locations)

elif action == "inspect":

if target == "inventory":

inspect\_inventory(inventory)

elif target == "room":

inspect\_location(current\_location, locations)

else:

print("Cannot inspect that")

else:

print("Invalid command. Try 'go [direction]', 'get [item]', or 'inspect [room/inventory]'.")

if current\_location == 'Throne Room' and 'crown' in inventory and 'sword' in inventory:

print("Congratulations! You have claimed the throne with the crown and the sword!")

break

if 'item' in locations[current\_location] and locations[current\_location]['item'] == 'dragon':

print("You found the dragon... Game Over!")

break

**The show\_game\_options function**

def show\_game\_options() -> None:

In this function you are supposed to print the instructions for the game. The output needs to look like the following:

|  |
| --- |
| The 'Knight in the Kingdom', a textual "Adventure Game"  ===================  Commands:  go [north|south|east|west] - Move in the specified direction  get [item] - Pick up an item  inspect room - See the items in the current room  inspect inventory - See the items in your inventory |

Ideally, to get full marks, you should print the entire text with a single print function call.

**The show\_status function**

def show\_status( current\_location: str, locations: dict ) -> None:

In this function you need to print the location in which the player character currently is and the directions in which the player can move, for instance:

|  |
| --- |
| You are in the Tower  Direction allowed are north or east or south |

**The get\_command function**

def get\_command() -> tuple:

As it is evident from the mainloop, the objective of this function is to request and parse the command provided by the player.

The function gets the command from the user as a single user input. Well-formed commands are composed by two strings separated by space, in accordance with the instructions provided by the show\_game\_options function.

The input received by the get\_command function is not meant to be case sensitive, i.e. commands from the user are accepted in both lowercase and uppercase format, with no distinction.

**The move\_to\_next\_location function**

def move\_to\_next\_location( current\_location : str, target:str, locations:list) -> str:

In this function, if the direction exists and is allowed from the current location the character moves to the next location, otherwise it remains in the current location.

**The pickup\_item function**

def pickup\_item(target: str, inventory: list, current\_location: str, locations:dict) -> None:

In this function, if the requested item exists and is in the current location, the item is picked-up and stored in the inventory. Obviously, the item needs to be removed from the dictionary, so that, if the player passes again in this location, the item is no longer there. If the item does not exist, this function prints on the screen:” Sorry, I do not see <item> here”.

Examples:

|  |
| --- |
| > get hammer  Sorry, I do not see hammer here |

**The inspect\_inventory function**

def inspect\_inventory( inventory: list ) -> None:

This function prints the content of the inventory.

**The inspect\_location function**

def inspect\_location( current\_location: str, locations: dict ) -> None:

This function prints the items found in the room. If there is an item in the room, the function prints “In this location I see <item>”, if there is no item in the room the function prints “There is nothing to see here”.

Examples:

|  |
| --- |
| > inspect room  In this room I see map |

|  |
| --- |
| > inspect room  There is nothing to see here |

If you win against the dragon, then the game continues, but you need to print a statement to celebrate the victory and the “dragon” item needs to be removed from the location dictionary, so that, if the player enters in the same room again, the dragon is no longer found.

**Calling the mainloop.**

The mainloop is a function, please write the entry point for the program that calls such a function, i.e. game\_mainloop. The player needs to start the game from the Tower.

**Enhancing the game**

* **Defeating the dragon.** In the current implementation of the game, if the dragon is found, the player immediately loses the game. Add instructions so that, if you have the sword item or the spell book in the inventory when meeting the dragon, you can defeat it, otherwise you will lose, and the game will be over.
* **Items in a random location.** Modify the code to randomly distribute the items in the locations (including the dragon). In each location only one item can be present.

**Commenting the code.**

Comment the code explaining what its purpose is, and the choices made.

**Example**

An example of run of the game is illustrated below.

|  |
| --- |
| The 'Knight in the Kingdom', a textual "Adventure Game"  ===================  Commands:  go [direction] - Move in the specified direction  get [item] - Pick up an item  inspect room - See the items in the current room  inspect inventory - See the items in your inventory    You are in the Tower  direction allowed north or east or south    >go north  You are in the Throne Room  direction allowed south    >inspect room  In this location I see crown  You are in the Throne Room  direction allowed south    >get crown  crown picked up!  You are in the Throne Room  direction allowed south    >inspect inventory  Your inventory contains: crown  You are in the Throne Room  direction allowed south    >go south  You are in the Tower  direction allowed north or east or south    >inspect room  In this location I see map  You are in the Tower  direction allowed north or east or south    >go east  You are in the Armory  direction allowed west or north    >inspect room  In this location I see sword  You are in the Armory  direction allowed west or north    >get sword  sword picked up!  You are in the Armory  direction allowed west or north    >inspect inventory  Your inventory contains: crown,sword  You are in the Armory  direction allowed west or north |

**Marks awarded**

|  |  |  |
| --- | --- | --- |
| **Description** | **Submenu** | **Marks** |
| **Implementation** |  | **80%** |
|  | **show\_game\_options** | **5%** |
|  | **show\_status** | **5%** |
|  | **get\_command** | **10%** |
|  | **move\_to\_next\_location** | **10%** |
|  | **pickup\_item** | **10%** |
|  | **Inspet\_inventory** | **10%** |
|  | **Inspect\_location** | **10%** |
|  | **mainloop** | **5%** |
|  | **Defeat the dragon** | **10%** |
|  | **Items in random locations** | **5%** |
| **Comments** |  | **5%** |
|  |  |  |
| **Code structure and format** |  | **15%** |

For generic marks description see Appendix A.

**Continuous Assessment Submission Guidelines**

**Artificial Intelligence**

As part of the assessment strategy, students may be called upon, at random, to orally defend any piece of work submitted. This would be conducted, not as a punitive measure, but rather as a way of positively supporting the overall assessment process and ensure that it is robust. Work submitted through Turnitin will produce an AI report as well as a Similarity report. Assignments which have copied work from websites, from other authors, from other students or any other sources may receive a grade of 0.

Assignments must be submitted via Turnitin. If assignments are not submitted via Turnitin, this will be regarded as a non-submission. It is the student’s responsibility to upload assignments to the correct Turnitin link.

Please observe the suggested word count. Assignments that are too short or too long may be penalised.

**Plagiarism**

Assignments which have copied work from websites, from other authors, from other students or any other sources may receive a grade of 0. All instances of plagiarism must be reported to the Head of Department who, in turn, is obliged to report them to the Registrar. Students who receive this grade may be asked to justify their actions to the University’s plagiarism panel. Students must apply to the examination board at the end of the academic year to re-submit their work in such instances. Please note that copying verbatim from original sources is unacceptable even if you provide references.

**Extension Policy**

Only in exceptional circumstances will extensions be granted.

Undergraduate and postgraduate extensions cannot be granted by your lecturer. Such extensions can only be granted by the Faculty of Lifelong Learning once a completed extension form and supporting documentation is returned online.

Students can apply for extensions at the following link - [LLL Extension Request Form](https://www.cognitoforms.com/SETU10/LifelongLearningExtensionRequestForm)

Extensions must be sought in advance of the submission date. Extensions will not be granted retrospectively.

The circumstances under which an extension request will be considered include, but are not limited to:

* Serious personal / family / business reasons
* Where a student is representing their country or university

An extension request is **unlikely** to be considered under the following circumstances:

* Minor illnesses such as a common cold
* Holidays during the academic year
* Multiple assignments due at the one time
* Failure to plan study schedule
* Debs / weddings / social events
* IT and /or computer failure

The Faculty of Lifelong Learning reserve the right to request supporting documentation. If you are applying for an extension that exceeds **5 days** you must submit supporting documentation (e.g. letter from a doctor, employer, line manager etc.) so that any prolonged absence can be verified.

To avoid any unnecessary distress, please ensure that you **back up your work** regularly as you undertake your assignment.

Failure to submit a piece of assessment may result in a grade of 0.

**Referencing**

***Correct referencing and citation are very important in academic writing.***

Please consult PACE MODULE 3 - [Citations and References](https://librarycarlow.setu.ie/library/citations-and-references/#/) for the referencing system used by SETU. Note ‘Credit Where Credit is Due’ is a supportive tool available in the Summary and Further Resources section.

**Appendix A.**

**Grade Descriptor for Scripting**

|  |  |  |
| --- | --- | --- |
| **Grade** | **Criteria relevant to assessing Knowledge, Understanding, Application** (Bloom’s levels 1–3) | **Additional criteria relevant to assessing Analysis, Synthesis, Evaluation** (Bloom’s levels 4-6) |
| **70 – 100**  **1.1** | **Excellent** A comprehensive, highly structured, focused and concise response to the assessment task, consistently demonstrating:   * An extensive and detailed   knowledge of the subject matter.   * Ability to analyse problems and identify requirements to provide an optimal solution * Addresses all of the specifications and provides an optimal solution * Applies all of the scripting concepts correctly * Clean code with no superfluous code. * Excellent presentation (code structure, documentation). Anybody can understand the code. | A deep and systematic engagement with the assessment task, with consistently impressive demonstration of a comprehensive mastery of the subject matter, demonstrating:   * An ability to identify correctly all input and output and provide alternatives * An ability to identify and use proper elements and concepts and clearly document their use * An ability to apply understand system requirements and produce correct results * The script works and meets all specifications. Does exceptional checking for errors and out of- range data * Understand and integrate with external libraries seamlessly * Documentation is well written and clearly explains what the code is accomplishing * The code is extremely well organized   and easy to follow and adheres to coding standards throughout |
| **60 – 69**  **2.1** | **Very Good** A thorough and well-organised response to the assessment task, demonstrating:   * A broad knowledge of the module matter. * Addresses all of the specifications and provides a correct solution * Applies nearly all of the scripting concepts correctly * No irrelevant or unnecessary code * Good presentation (code structure, documentation). Any expert could understand the code | A substantial engagement with the assessment task, demonstrating:   * An ability to identify correctly all input and outputs * An ability to identify and use proper code elements and concepts * An ability to apply requirements and produce partially correct results. * Understand and integrate with external libraries * A script that runs and meets all specifications, does some error checking * Documentation is simple comments and header that is useful in understanding the code * The code is well organized and adheres to coding standards |
| **50 – 59**  **2.2** | **Good** An adequate and competent response to the assessment task, demonstrating:   * Adequate but not complete knowledge of the module matter. * Addresses most of the specifications and provides a mostly correct solution * Applies most of the scripting concepts correctly * Very little irrelevant or unnecessary code * Good presentation (code structure, documentation). An expert could understand the code with some extra explanation | An intellectually competent and factually sound answer with, marked by:   * An ability to correctly identify some input and outputs * An ability to identify some correct elements and concepts and applied correctly * An ability to apply requirements but does not produce correct results * Integrates with external libraries. * The script produces correct results but does not display correctly, does little checks for errors * Documentation is a few simple comments that explain the code. * Code is readable and adheres to some coding standards |

|  |  |  |
| --- | --- | --- |
| **40 – 49**  **Pass** | **Satisfactory** An acceptable response to the assessment task with:   * Basic grasp of module matter, but somewhat lacking in focus and structure. * Addresses some of the specifications and provides a partially correct solution * Applies some scripting concepts correctly * A small amount of irrelevant or unnecessary code. * Satisfactory presentation (code structure, documentation) with an acceptable level of presentation errors. | An acceptable level of intellectual  engagement with the assessment task showing:   * An ability to identify only one input or output * An ability to identify code elements and concepts but not applied correctly * An ability to identify requirements but not applied correctly * Integrates with external libraries poorly * The script produces correct results but does not display correctly, no checks for errors. * Documentation is a few simple comments that do not explain the code well * Code poorly organized and does not adhere to coding standards |
| **0 – 39**  **Fail** | **Unacceptable** A response to the assessment task that is unacceptable, with:   * A clear lack of understanding of the subject matter * A failure to address specifications and provide a correct solution. * Cannot apply scripting concepts correctly * Incomplete or broken code * A large amount of irrelevant or unnecessary code * Poor presentation (code structure, documentation) * Evidence of substantial plagiarism | An unacceptable level of intellectual engagement with the assessment task, with:   * An inability to identify any inputs or outputs * An inability to identify code elements and concepts * An inability to identify requirements * An inability to use external libraries * The script produces incorrect results * No documentation * Code is completely unorganized and ignores all coding standards |

2 Anderson, L. W. and David R. Krathwohl, D. R., et al (Eds..) (2001) *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Allyn & Bacon. Boston, MA

Mustapha, Aida & Samsudin, Noor & Arbaiy, Nureize & Mohamed, Rozlini & A Hamid, isredza rahmi. (2016). Generic assessment rubrics for computer programming courses. 15. 53-61.