Unit:
Introduction to Programming with Python

Assignment title:
Snakes and ladders

[Cycle] [Year]

Important notes

- Please refer to the *Assignment Presentation Requirements* for advice on how to set out your assignment. These can be found on the NCC Education website. Hover over 'About Us' on the main menu and then navigate to 'Policies and Procedures' then scroll to the 'Student Support' area.
- You **must** read the NCC Education document *Academic Misconduct Policy* and ensure that you acknowledge all the sources that you use in your work. These documents are available on the NCC Education website. Hover over 'About Us' on the main menu and then navigate to 'Policies and Procedures' then scroll to the 'Student Support' area.
- You **must** complete the *Statement and Confirmation of Own Work*. The form is available on the NCC Education website. Hover over 'About Us' on the main menu and then navigate to 'Policies and Procedures' then scroll to the 'Student Support' area.
- Please make a note of the recommended word count. You could lose marks if you write 10% more or less than this.
- You must submit a paper copy and digital copy (on disk or similarly acceptable medium). Media containing viruses, or media that cannot be run directly, will result in a fail grade being awarded for this assessment.
- All electronic media will be checked for plagiarism.

Introduction

Plan, create and test a Python program to meet the given scenario.

You need to produce a working Python program and a Word document containing the following:

- Success criteria
- Decomposition of the problem
- Design of modules and algorithms using pseudocode
- The Python program code
- Screenshots of the Python program working
- A test log including a range of tests and screenshot evidence of the results
- Installation and user guide documentation

Scenario

Snakes and ladders is a board game. The board is made up of 10 x 9 squares. The user starts in the bottom left hand corner and has to move to the top right corner.

The board has ladders and snakes. If the user lands on the bottom of a ladder, they move to the top of the ladder. If the user lands on the top of a snake, they move to the bottom of the snake.
For this program, these will be replicated by a 'jump x' for ladder (where x is the position the piece moves forward to) and a 'go back x' for a snake (where x is the position the piece moves backwards to).
Here is an example board layout, if a user's piece stops on position 7 then they move forward to position 34:

81	82	83	84	85	86	87	Go back 28	89	90
80	79	78	77	76	75	74	73	Jump 81	71
61	62	63	64	65	66	67	68	69	70
Go back 3	59	58	57	Jump 76	55	54	53	52	51
41	42	43	44	45	46	47	48	49	50
40	39	38	37	36	35	34	Go back 18	32	31
21	22	23	24	25	26	27	28	29	30
20	19	18	17	16	15	14	13	12	11
Start	2	3	4	5	6	Jump 34	8	9	10

Two users take it in turns to roll two dice (1-6) and move their piece forward the number of spaces rolled.

If the user's piece stops on a 'jump' they move forward to the position identified.
If the user's piece stops on a 'go back' they move back to the position identified.

The first user to get to the final space is the winner.

The position and content of the Jump and Go back spaces are stored in an external text file that needs to be read into the program at the start of the program. An example of the format of this file for the given board is:

```
7,Jump 34
33,Go back 18
56,Jump 76
60,Go back 3
72,Jump 81
88,Go back 28
```

Task – 100 Marks

Task 1 Decomposition and design – 40 marks

- Identify a list of success criteria for the program that you will compare your finished program to.
- Decompose the problem into subproblems. Create one or more structure diagrams for the problem.
- Design algorithms for the subproblems, for example functions, using pseudocode.

Task 2 Implementation – 30 marks

- Create your program using Python
- Test your program regularly to make sure it works
- Create a copy of your program code and include screenshots to show the implementation of your system.

Task 3 Testing – 25 marks

- Creating a testing strategy for your program
- Include tests for all aspects of the program
- Include a range of test data include normal, extreme and invalid
- Test your program using your strategy and produce evidence of the outcomes using a test log
- Complete the results of each test
- Include evidence for each test
- Test your program against your success criteria. Produce evidence, or refer to where the evidence can be found, for the results of your testing.

Task 4 Publishing programme and installation documentation – 5 marks

- Create an installation guide for your program
- Create user documentation for your program to explain how to start, play and close the game.

Mark distribution

- Design (40 marks)
- Implementation (30 marks)
- Testing (25 marks)
- Publishing programme and installation documentation (5 marks)

Hints on design and implementation

- Your program does not need to make use of graphics, it can provide a text-output of the board.
- Your board can be stored as a 1D array and then the output of it can format it in the required way, or it can be a 2D array that is stored as shown.
- Object-oriented programming may be useful for the creation of the board.
- Include appropriate exception handling when reading from the text file.

Guidance

The assessment of your project will depend in part upon the quality of the documentation that you have produced.

- Restate the specification of the assignment by listing the Required Outcomes.
- Take the time required to design the assignment before you type any program code. Follow an appropriate design and documentation sequence.
- Always document your designs **before** you implement them.
- Provide a detailed design including, where appropriate, the design of any algorithms.
- Build in error handling to involve meaningful messages that would help with any future maintenance of the software.
- Annotate all implementation.
- Design a testing strategy.
- Justify the design of suitable comprehensive test data.
- Show evidence of testing.
- Where appropriate, detail any major corrective action that you have taken in the light of the testing process.

Submission requirements

A word-processed document must be submitted incorporating the full documentation of all the significant aspects of the development of the assignment above. The document should be submitted both in paper form and digital form.

Refer to the Guidance above when producing your final documentation.

You are required to submit a publishable copy of the compiled system together with installation notes.
This publishable copy, that includes a setup file, should be on an appropriate medium (CD, USB flash drive, etc.).

Candidate checklist

Please use the following checklist to ensure that your work is ready for submission.

Have you read the NCC Education document *Academic Misconduct Policy* and ensured that you have acknowledged all the sources that you have used in your work?	❑
Have you completed the *Statement and Confirmation of Own Work* form and attached it to your assignment? **You must do this.**	❑
Have you ensured that your work has not gone over or under the recommended word count by more than 10%?	❑
Have you ensured that your work does not contain viruses and can be run directly?	❑