COMP9021 Principles of Programming
Term 2, 2024

Assignment 2

Worth 13marks and due Week 11 Monday @ 10am

1. General Matters

1.1 Aim

The purpose of t

- Develop y
- Design ar hon program.
- Analyse t s basic elements into num
- Check tha
- Output th
- Use **objec**

1.2 Marking

This assignment

Ma

ks

```
display_features() method                    10.0 marks
    gates                                 1.5
    walls that are all connected          1.5
    inaccessible inner points             1.5
    accessible areas                      1.5
    accessible cul-de-sacs                2.0
    entry-exit paths                      2.0

Total                                     13.0 marks
```

Your program will be tested against several inputs. For each test, the auto-marking script will let your program run for **30 seconds**. The outputs of your program should be **exactly** as indicated.

1.3 Due Date and Submission

Your programs will be stored in a file named **labyrinth.py**. The assignment can be submitted more than once. The last version just before the due date and time will be marked (unless you submit late in which case the last late version will be marked).

Assignment 2 is due **Week 11 Monday 5 August 2024 @ 10:00am** (Sydney time).

Please note that **late** submission with **5% penalty per day** is allowed **up to 5 days** from the due date, that is, any late submission after **Week 11 Saturday 10 August 2024 @ 10:00am** will be discarded.

Please make sur **[Mark]** button in **Ed**. It is your resp **bmissions** link in Ed otherwise you

1.4 Remind

You are permitte people. Such discussions must **on on your own**. Submissions are people's work or work very closely

2. Descriptio

The representati d **3** such that:

- **0** codes rs:
- **1** codes ones:
- **2** codes t ones:
- **3** codes

A **point** that is connected to **none** of their **left**, **right**, **above**, and **below** neighbours represents a **pillar**:

Analysing the labyrinth will also allow to represent:

- **cul-de-sac:**
- **entry-exit path:**

3. Examples

3.1 First Example

The file named **labyrinth_1.txt** contains the following:

```
1 0 2 2 1 2 3 0
3 2 2 1 2 0 2 2
3 0 1 1 3 1 0 0
2 0 3 0
3 2 2 0
1 0 0 1
```

As per the codin

Here is a possibl

```
$ python3
...
>>> from labyrinth import *
>>> lab = Labyrinth('labyrinth_1.txt')
>>> lab.display_features()
The labyrinth has 12 gates.
The labyrinth has 8 sets of walls that are all connected.
The labyrinth has 2 inaccessible inner points.
The labyrinth has 4 accessible areas.
The labyrinth has 3 sets of accessible cul-de-sacs that are all connected.
The labyrinth has a unique entry-exit path with no intersection not to cul-de-sacs.
>>>
```

3.2 Second Example

The file named **labyrinth_2.txt** contains the following:

```
022302120222
222223111032
301322130302
312322232330
001000100000
```

As per the coding above, **labyrinth_2.txt** will look like the following:

Here is a possible

```
$ python3
...
>>> from lab
>>> lab = La
>>> lab.disp
The labyrint
The labyrint
The labyrinth has 4 inaccessible inner points.
The labyrinth has 13 accessible areas.
The labyrinth has 11 sets of accessible cul-de-sacs that are all connected.
The labyrinth has 5 entry-exit paths with no intersections not to cul-de-sacs.
>>>
```

3.3 Third Example

The file named **labyrinth_3.txt** contains the following:

```
31111111132
21122131202
33023022112
20310213122
31011120202
21230230112
30223031302
03122121212
22203
221103
111111
```

As per the codin

Here is a possibl

```
$ python3
...
>>> from labyrinth import *
>>> lab = Labyrinth('labyrinth_3.txt')
>>> lab.display_features()
The labyrinth has 2 gates.
The labyrinth has 2 sets of walls that are all connected.
The labyrinth has no inaccessible inner point.
The labyrinth has a unique accessible area.
The labyrinth has 8 sets of accessible cul-de-sacs that are all connected.
The labyrinth has a unique entry-exit path with no intersection not to cul-de-sacs.
>>>
```