# dSchool of Computing

| | |
|---|---|
| Module Code | M30299 |
| Module Title | Programming |
| Module Coordinator<br><br>Lecturer for this coursework | Matthew Poole<br>matthew.poole@port.ac.uk<br><br>Nadim Bakhshov<br>nadim.bakhshov@port.ac.uk |
| Assessment Item Number | Item 3 |
| Assessment Title | Java coursework: A Text-based Story |
| Date Issued | Friday 1 April 2022 |


**Schedule and Deliverables**

| **Deliverable** | **Value** | **Format** | **Deadline** |
|---|---|---|---|
| One ZIP Moodle Upload | 100%<br><br>25% of module | a .zip file to contain:<br>- .pdf document of graphical story map<br>- folder with project files inside*:<br><br>All files including the .zip should be named with student ID<br><br>**It is imperative that you test the project works. The Project must work with intelliJ (JDK 15) for the demonstration. If you do the challenge you must discuss how the demonstration will be carried out with nadim.bakhshov@port.ac.uk* | 11.00pm, Friday 6th May 2022<br><br>**Late/ECF deadline:** 11:00pm, Friday 20th May 2022 |

## Notes and Advice

- The Extenuating Circumstances procedure is there to support you if you have had any circumstances (problems) that have been serious or significant enough to prevent you from attending, completing or submitting an assessment on time. If you complete an Extenuating Circumstances Form (ECF) for this assessment, it is important that you use the correct module code, item number and deadline (not the late deadline) given above.
- ASDAC are available to any students who disclose a disability or require additional support for their academic studies with a good set of resources on the ASDAC moodle site
- The University takes any form of academic misconduct (such as plagiarism or cheating) seriously, so please make sure your work is your own. Please ensure you adhere to our *Code of Student Behaviour* and watch the video on Plagiarism.
- Where appropriate, any material included in your coursework should be fully cited and referenced in **APA 7** format. Detailed advice on referencing is available from the library, also see TECFAC 08 Plagiarism.
- Any material submitted that does not meet format or submission guidelines, or falls outside of the submission deadline could be subject to a cap on your overall result or disqualification entirely.
- If you need additional assistance, you can ask your personal tutor, student engagement officer ana.baker@port.ac.uk, academic tutor xia.han@port.ac.uk or your lecturers.
- If you are concerned about your mental well-being, please contact our Well-being service.

# M30299 - Programming

moodle.port.ac.uk

## Java Coursework: A Text-based Story

## Introduction

This coursework assignment is designed to give you practice in combining and applying all of the main programming concepts you've seen in the Java section of the module to solve a more complex problem. The assignment will be marked out of 50 and carries 25% of the module marks (it is 100% of assessment Item 3 of the module.)

You need to submit your program via the module's Moodle site by the deadline of ***11.00pm, Friday 6th May 2022***, and are required to ***demonstrate*** your submitted program in your 2-hour Programming class timetabled during the week beginning ***9th May 2022***. Study this handout thoroughly in order to understand exactly what is expected from you for the coursework.

## Tasks

## Task 0 - Create Project

Before starting individual tasks you should build a fresh Java project using your studentID as the project name. For further guidance see step 1 of How to build an interactive story/game

## Task 1 - Design Story Map

Design a story map. It should be a text-based story with a minimum of approximately 40 *yes/no*[1] *user based decisions*. You should build a graphical picture of your story map. On the next page is a sample to provide you with a rough template. You can use this to guide you on the number of decision points in your story. I would recommend you also watch *Bandersnatch* (the Black Mirror Episode) or seek out those 1980 interactive books, or simple text based console games. The example was chosen because it is, as an example, light and quite easy to understand. **You cannot use this example in your own coursework. It is only provided as guidance**. Historically it was actually turned into a

[1] *This is recommended but you are allowed to have more options at each point in your story. Please be aware of the increased complexity of your code for more options.*

book generated by a large 1960s mainframe computer. The story starts at the middle of the top of the map. It has several weaknesses which you should try and avoid:

| **Shortcoming** | **Details/Comments** |
|---|---|
| Infinite dead end loops | If you follow the logic through you will find some dead-end in this interactive story map. A dead-end is a place in the interactive story map that gets stuck in a loop that you cannot escape from. Try to avoid these. |
| Illogical decision movement | There are some strange movements between decisions. sudden jumps in logic and flow. Try to make your story flow. |

See full image of sample being used
https://drive.google.com/file/d/1N359pWu_WTQZd7Ffu5KgiiIgxqtgKNz6/view?usp=sharing

We list some suggestions below.

## Suggestions for your story map

| **Idea** | **Details/Comments** |
|---|---|
| Variation on the Perec scenario | You can build a variation on the given **Perec** scenario. If you are going to do a variation then you must ensure the scenario is interesting and rich enough to be worth tackling.<br><br>With Perec you will see the whole map turns on the idea: **the art of asking your boss for a raise**. It is, of course, not entirely serious. It is slightly comical and built to be entertaining. It must be very specific, not vague or general.<br><br>Some (comical) suggestions:<br>**the art of asking your friend for money for milk**<br>**the art of discussing an awkward event at last night's party**<br>**etc.** |
| Labyrinth/Maze Variations | The idea behind a labyrinth type map is to create any kind of landscape that has junctions and paths which your user can travel through. For example, a simple retro text based adventure game uses a labyrinth structure to map out a landscape a character moves through. Or you can attempt to build something more complex and animated.<br><br>The concept is to turn a place and related surroundings into a decision based structure. For example, if you designed a game to get to a particular location then the labyrinth decision structure would map out all the junctions where you can go different ways, make different decisions in encountering characters along the way or even objects. |

| | |
|---|---|
| | **Some suggestions:**<br>Escape rooms/cities/places<br>Dungeons & dragons' type game<br>Shoot-em-up - with landscape mapped with paths and enemies placed along certain paths.<br>etc |
| You can borrow from other sources | As your application is for educational purposes you should generally be okay to borrow material (reference it in your codebase). |
| Speak to the team for any further possible ideas | If you struggle to find a story or do not feel creative then consider chatting to nadim.bakhshov@port.ac.uk about alternatives. |

See the handout "CW - Story Map" for further guidance.

For further guidance see step 2 of How to build an interactive story/game

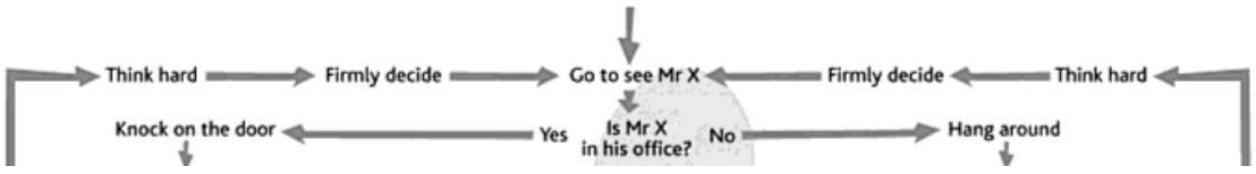
## Task 2 - Translate Map to .csv

Translate your graphical map into a flatfile data format: *.csv* format.

(*Alternative data formats are permitted: .json or .xml but before you attempt an alternative ensure you have discussed with nadim.bakhshov@port.ac.uk all the issues for your Java code. Throughout this handout we will refer to csv. If you have agreed an alternative format then assume you should replace csv in the handout with your agreed data format*)

It is not recommended, nor required, for you to build a back end database using any specific database platform.

Each user interaction point in the above map will be translated into a line of the .csv file. To do this you should **number** each node (decision point). You then link the decisions at each point to other decision points. For example,




We mark '**Go to see Mr X, Is Mr X in his office?**' as **node 0**. The **'Yes'** option takes you to '**Knock on the door**' which is **node 1**. The **'No'** option takes to '**Hang Around**' which is **node 21**. Here is a partially completed .csv file for the given map:

```
0,1,21,Go to see Mr X,Is Mr X in his office?
1,2,31,Knock on the door,Does he look up?
2,32,3,-,Affirmatively?
3,4,31,-,Are you told to come back at 2.30?
4,6,5,Go back to your desk,Is today Friday?
5,6,7,-,Is it Lent?
6,9,8,Find out cafeteria lunch menu,Was fish on?
...
```

***Note***: "***Go to see Mr X***" *and* "***Is Mr X in office***" *are separate nodes and are replaced with just one node/line. Where appropriate you might wish to do this for your own maps.*

The numbering system is arbitrary and you can number each node in any way you like as long as you use a consistent numbering system. These numbers act as decision **nodeIDs**. For a specific line it follows the following structure:

e.g. `0,1,21,Go to see Mr X,Is Mr X in his office?`

| | |
|---|---|
| ***nodeID*** | `0` |
| ***yesID*** | `1` |
| ***noID*** | `21` |
| ***description*** | `Go to see Mr X` |
| ***question*** | `Is Mr X in his office?` |

To create a .csv file you should either (a) manually type each row into a plain text file and save the file with the correct file or (b) use a spreadsheet/google sheet and output the result as a .csv file. Here is a sample spreadsheet building the data.

| 0 | 1 | 21 | Go to see Mr X | Is Mr X in his office? |
|---|---|---|---|---|
| 1 | 2 | 31 | Knock on the door | Does he look up? |
| 2 | 32 | 3 | - | Affirmatively? |
| 3 | 4 | 31 | - | Are you told to come back at 2.30? |
| 4 | 6 | 5 | Go back to your desk | Is today Friday? |
| 5 | 6 | 7 | - | Is it Lent? |
| 6 | 9 | 8 | Find out cafeteria lunch menu | Was fish on? |
| 7 | 36 | 36 | - | Is today Monday? |

Once the spreadsheet has been constructed you can export the sheet as a .csv file.

For further guidance see step 3 of [How to build an interactive story/game]

## Task 3 - Build Java Code

**A brief overview**: Your program should load the *.csv* file into an appropriate combination of data structures and objects. The .csv file should then be closed. The console should display the first step of your story that starts the interactive story. Using the console the user should be able to navigate through the interactive story. Depending on the decision made by the user the program should correctly follow the path in the interactive story map. The program should navigate through the relevant path until it reaches another decision. At each decision point the console should ask the user a question and offer a minimum of two options: *yes/no*. When the user selects one of these options it should take the selected path to the next decision point. In this way the user should then be able to navigate through the story, making decisions at each relevant point.

**Details:** It is suggested your code carries out the following steps:

1. Open a link to your csv file. Set up a loop which checks if the csv file has a line of data.
    a. Read the next line from the file
    b. Split each line (on the commas).
    c. Design and build a Node object with correct data types for each item of the split item i.e all numerical IDs used in the navigation should be stored as integers - not string data.
    d. Add each Node object to an array or array list.
2. To navigate your console interface must display appropriate data for each decision point of the interactive story and allow user input to select the option for navigating through the story.

Here is a sample from an interactive story to illustrate the console interactions for your finished program.

```
Go to see Mr X
Is Mr X in his office?
Yes or No? (press 1 for Yes or 2 for No)
> 1
Knock on the door
Does he look up?
Yes or No? (press 1 for Yes or 2 for No)
> 2
Affirmatively?
Yes or No? (press 1 for Yes or 2 No)
> 1

etc.
```

Consider the following section:

```
Go to see Mr X
Is Mr X in his office?
Yes or No? (press 1 for Yes or 2 for No)
> 1
```

This is made up as follows:

| | |
|---|---|
| `Go to see Mr X` | Description - this text describes where you are in the story. |
| `Is Mr X in his office?` | Question - this text displays the decision for the user to make |
| `Yes or No? (press 1 for Yes or 2 No)` | Standard text attached to each decision - The story map assumes a maximum of two options at each decision point in the interactive story. |

The text at the prompt

```
> 1
```

is input text from the user at the console.

The User interface must conform to the following points:

1. You cannot enter any other number for the navigation other than the specified options
2. You must not use the actual Node IDs for navigation. i.e. If the yes option is Node 1 and the no option is Node 22 then you should not use 1 or 22 in the navigation.
3. Your user interface must not crash. It must warn the user the option is incorrect and repeat the description, question and options.

For further guidance see steps 4 - 9 of How to build an interactive story/game

## Task 4 - Optional Challenge task

The above requirements are what I expect most students to attempt, and carry the vast majority of the marks for functionality. If you would like a further challenge for additional marks, then I encourage you to attempt this additional feature.

Using Java AWT/Swing or Android Studio create a GUI to replace the console application with the use of appropriate controls and text in a mobile sized window/interface.

## Moodle Submission

You should submit your single zip file incuding all your work:

- graphical story map (.pdf file) using student number i.e. 2001234.pdf
- program(s) folders (including Java projects in separate folders)

in a single .zip file via the module's Moodle site by the deadline specified above. Make sure that your overall submission is named using your student number, and that it has a .zip suffix; for example, 2001234.zip. Click on the link labelled Java Assignment in the Assessment tab and upload your program. If you miss the deadline, you will need to upload it using the late Submissions link and your mark will be capped according to University regulations.

It is critical that you test your submission and ensure it can be run during the demonstration period. The project must run on IntelliJ IDEA with Java SDK 15 or above. If you attempt the challenge then you must discuss the demonstration process with nadim.bakhshov@port.ac.uk.

## Demonstration

You need to demonstrate your program to a member of staff in your Programming session timetabled during the week beginning 9th May. We will execute your submitted program, and we will ask you questions about how you wrote it and how it works. During the demonstration we will establish the level of understanding of your code. ***If you are unable to answer questions about your code then you may lose marks.*** All the marks for the assignment will be awarded during the demonstration, so you must attend: failure to attend the demonstration will result in zero marks, and demonstrating your program late will result in your mark for the assignment being capped under University rules. If you wish to organise a late demonstration outside a timetabled session, please email me—you must have given your demonstration by Friday 27th May or you will receive a mark of zero. Formal written feedback and your assignment mark will be sent to you via email after your demonstration has been completed. If you do not receive this email, then your mark may not have been recorded and it is your responsibility to inform me if this happens.

## Core Console Based Application [40 marks]

In the demonstration we will first assess the completeness and correctness of the core operation of your program, and the quality and robustness of its user interface. We will follow a number of paths, using your graphical map for guidance. *If you have attempted the GUI (challenge feature) then the* ***core functionality*** *can be demonstrated through the GUI.*

These marks will be given as follows:

| | |
|---|---|
| **Functionality**<br><br>*This will be largely determined by the correspondence of the **navigation** to the **story map**. Your story map will support the grading of functionality. We will compare then navigation through your map with actually executing your software. For your demonstration you will be expected to show different pathways through your map.*<br><br>*Ensure your map*:<br>1. Uses approximately 30-40 nodes. It can be larger.<br>2. Has significant alternative pathways | 25 |
| **Code Decisions, Design & Quality**<br><br>*This will **involve a conversation with the student on coding decisions and choices**. An inability to answer questions will affect the grading of code quality.*<br><br>You might be asked about any of the following:<br><br>1. Your use of a .csv to hold your data and your file loading<br>2. Your file reading mechanisms<br>3. Use of a Node object or alternative to take a line from the file<br>4. Use of Array or ArrayList to store Nodes (map)<br>5. How your navigation mechanisms work | 15 |

# Challenge Feature [10 marks]

If you attempt the challenge then they have an option to gain the **core functionality marks** through the GUI navigation.

These marks will be given as follows:

| | |
|---|---|
| **Functionality**<br><br>*This will be determined by the correspondence of the navigation to the story map and the usability of the graphical user interface.* | 10 |

## General advice

Most importantly, start early and do not leave the coursework until just before the deadline. Your work will almost always suffer if you leave it until too late. Furthermore, technical problems are likely to be overcome if encountered early, and do not usually constitute an acceptable reason for lateness. If you find the task very difficult, remember that you do not have to provide a complete solution to achieve a pass mark. Make sure that your program executes and can display the contents of the .csv file through an appropriate data structure. If you don't know how to start, it is recommended that you see Nadim Bakhshov, Xia Han or Eleni Noussi for some advice as soon as possible.

## Hint

If well designed, your program will consist of interacting objects, each managing different parts of the program. It is suggested each decision in your story is managed in its own object (node) and that a data structure (array, arraylist, etc) manages the collection of node objects.

## Support

Queries should be addressed to me by email (nadim.bakhshov@port.ac.uk), although please note that I will be unavailable during some of the Easter break.

You can raise points using the faq form and all answers will appear here. Please check there before asking a question. The last few practicals before the deadline can be used for help on the coursework. The Academic Tutors, Xia Han and Eleni Noussi, are also able to give advice on the coursework.

## Important

This is individual coursework, and so the work you demonstrate and submit for assessment must be your own. Any attempt to pass off somebody else's work as your own, or unfair collaboration, is plagiarism, which is a serious academic offence. Any suspected cases of plagiarism will be dealt with in accordance with University regulations.

Nadim Bakhshov
February 2022