CCNU-UOW

CSCI851 Advanced Programming
Autumn 2020

Prof. Zhifeng Wang

Assignment 2 (Worth 10%)

Due 11:55pm 23^{th} November 2020.
Early diagram submission: 7pm 18^{th} November 2020.

Overview

This assignment is to be implemented using object oriented programming. It involves implementing a simulation of a journey of space exploration.

In addition to providing code you need to provide a pdf report explaining:

1. You need to submit a UML like class diagram indicating how your classes are related. A draft of this should be submitted by the earlier deadline.
2. You need to explain the ranges of the characteristics for your spaceship builds and officers.
3. You need to explain the alien civilisations that you set up.
4. You need to explain how the results of interactions between entities within events are determined, including battles. You can use diagrams to help to do this, but you don't have to.

General notes

These are some general rules about what you should and shouldn't do.

1. Your assignment should be sensibly organised with the same kind of expectations in this area as assignment one. No memory leaks etc. too.

2. **Other than the initial command line input, the program should run without user input. In particular this means there shouldn't be pauses waiting for the user to press a key.**

3. **There shouldn't be pauses generally, please don't use sleep!**

4. Your code must compile on Banshee with the compilation instructions you provide in `Readme.txt`. If it doesn't you will likely receive zero for the coding part of the assignment.

5. You have to use classes, and should make use of encapsulation.

6. You should make use of inheritance and, if appropriate, polymorphism.

7. If you use overloading, it should be sensible and appropriate overloading.

8. You can use your own data files for names or similar information.

1 A journey of space exploration

A crew of space explorers travel through space exploring planets, trading goods, and battling spaceships. The general overview is described below, but most of the details are up to you.

Your code should compile on Banshee according to instructions you provide in a `Readme.txt` file.

Once your program is compiled into the executable `JSE`, it must run as follows:

```
./JSE n
```

where n is the maximum number of sectors of space the journey will run for. The value of n should be between 1 and 50 inclusive. For various reasons, death and/or destruction mostly, the journey may not last for the specified number of sectors. There should be a fairly high chance of survival for journeys of up to 20 sectors so you may need to modify parameters to make this likely.

In each sector exactly one of the following will happen, with the chance as listed:

1. 35%: Spaceship encounter.

2. 25%: Planet encounter.

3. 15%: Trading station.

4. 15%: Empty sector. This is pretty boring.

5. 10%: Some other space thing. This is up to you.

Travelling between sectors costs resources. The primary resources that are used up are fuel, food, and money. You can choose the currency to be used. If the spaceship runs out of food or fuel the journey is over.

At the end of journey, a summary should be provided. This should include a statement of how resources have changed, including the final state, and a description of events. The number of crew who have died should be stated and there should be a list of dead officers, along with their former roles, death date/location, and cause of death.

2 Simulation components

The simulation is from the perspective of a spaceship and the associated crew. You need to give consistent and appropriate definitions for a class hierarchy associated with the people in the simulation. You also need to design appropriate classes to represent the planets, trading stations, and spaceships within the simultation.

2.1 The crew

The crew consists of officers and other ranks. Each of the officers influences particular activities that may be carried out.

- **Captain**: Negotiation which includes diplomacy and trading.
- **Pilot**: Influence travel efficiency, combat maneuvers, conflict evasion.
- **Engineering**: Influence travel efficiency, combat maneuvers, conflict evasion, defensive performance, systems recovery.
- **Mining**: Mining ability.
- **Weapons**: Offensive performance, defensive performance.

The individual abilities of officers should be randomly chosen but initially fairly high across those areas. If an officer dies a crew member should be promoted to that role. The new officer should have skills on average lower than the previous officer in that role. Officers should have names and ages, crew members don't have to be specifically identified and a simple count of them can be kept. Some crew members are likely to die too!

Those various activities are loosely described below. You need to think about how you are going to implement such activities.

- **Diplomacy**: Helpful in avoiding conflict.
- **Trading**: Selling and buying, including food and fuel.
- **Travel efficiency**: Fuel used for various movement activities.
- **Combat maneuvers**†: Avoid enemy fire while in combat.
- **Conflict evasion**: Running away in your spaceship.
- **Defensive performance**: Performance of ship shields and armour.
- **Systems recovery**†: Ability of parts of the ship to recover from damage/wear.
- **Mining ability**†: Ability to extract raw materials, and process them into goods that may be used or sold, including food, fuel, and tradable goods.
- **Offensive performance**: Performance of weapons systems.

The total number of crew should have some influence on some of those activities marked with a †. Those properties are also functions of the spaceship being travelled in.

2.2 The spaceship

The spaceship itself should be randomly chosen to be one a of at least five different spaceship builds that you have defined. They may vary through such abilities as cargo capacity, combat aptitude, mining, travel efficiency or speed. This doesn't mean you need different classes for the different spaceships, all of them likely have all abilities but to different degrees. Different instances from the same spaceship build should vary within ranges.

On initial setup the spaceship build, spaceship name and characterstics should be specified, as should the names and abilities of all officers. General resource and crew information should be provided too.

2.3 The events

Spaceships, planets, and trading stations may bring the explorers into contact with other civilisations.

You need to have at least five different types of alien species. Those species should have advantages/disadvantages associated with their abilities to perform various actions. For example, you may have a species that cannot trade, and another that cannot fight.

Trade is possible with a planet, or spaceship, or trading station. Conflict isn't possible with trading stations, but is possible with planets or spaceships. Mining is possible on planets.

Battles only need to be at the level of the spaceship. **There should be some sense of time within the simulation of conflict, so the battle should take place in stages or rounds rather than the result just being specified.**

Changes in the state of resources should be reported at the end of event, along with the values of the current state. Note that resource usage is dependent on such parameters as the number of crew. The spaceship can be repaired during travel in accordance with the systems recovery ability of the engineer.

Notes on submission

Submission is via Moodle.

Your code must compile on Banshee with the instructions you provide. If it doesn't you will likely be given zero for the programming part of this assignment.

Please submit your source, so .cpp and .h files, Readme.txt file, report, and makefile if you have one, directly to Moodle. There shouldn't be other files or directories and they shouldn't be in a zip file. Make sure your report is in pdf and has your name, student number, and lab class marked clearly on it.

A final diagram should be in your final report. Your early diagram submission should be in pdf, and should also have your name, student number, and lab class on it.

1. Late submissions will be marked with a 25% deduction for each day, including days over the weekend.
2. Submissions more than three days late will not be marked, unless an extension has been granted.
3. If you need an extension apply through SOLS, if possible **before** the assignment deadline.
4. Academic misconduct is treated seriously. Students involved will likely receive zero.