Airport simulation project

# Introduction

## 1.Program compilation

* Visual Studio 2022 Preview
* C++ environment

## 2. Linking

https://moodle.tuni.fi/course/view.php?id=21010

## 3. Installation

* Install g++
* Download file airport simulation project from Moodle.
* Find the code from Moodle
* Set up the code from all file

## 4. Running

* Compile every file from the directory
* Type command g++ \*cpp – o p to begin the program
* There are two modes to run the program

1. Step-by step:

Type p -p

1. Directing according to the point requirements

The details will shown in the next part of this report.

## 5. Work hours

|  |  |  |
| --- | --- | --- |
| Date | Work name | Implementation time |
| 08/11/2021 | Write the initial program. Think about the requirement and make some ideas. | 3 hours |
| 15/11/2021 | Arrange all the file to do the first part of projec P1 | 4.5 hours |
| 17/11/2021 | Complete P2,P3,P4,P5 | 3 hours |
| 19/10/2021 | Testing program | 2 hour |
| 21/11/2021 | Finish the program and write Readme.docx | 3 hours |

Total hour: 15.5 hours

## 6.Contact number

Email: [hung.truong@tuni.fi](mailto:hung.truong@tuni.fi)

Tel: 0403252908

7. Aiming score: 5/5

# Testing- Compilation proof

* Type p.exe on CMD
* Text

  Description automatically generated

## 1.Run P1 version of Airport simulation

Text

Description automatically generated

The result:

Text

Description automatically generated

Text

Description automatically generated

Replied to some question in this part:

* The maximum size of each queue is increased 🡪 the resulting values will be larger.
* The maximum size of each queue is decreased 🡪 the resulting values will be smaller.
* The size of the queue is increased 🡪 more planes can wait.

## 2.Run P2 version of Airport simulation

Text

Description automatically generated

The results:

Text

Description automatically generated with medium confidence

Text

Description automatically generated

Text

Description automatically generated

## Compare the number of planes between the two methods:

## The total of planes is equal to the two methods.

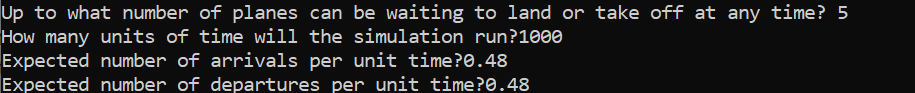
## The number of planes served successfully will be slightly less than double that of the basic one-runway airport, because, depending on the input parameters, one of the runways will sometimes be idle.

## The time when we implement the simulation 2 method is less than when we implement simulation 1.

## 3.Run P3 version of Airport simulation

A screenshot of a computer

Description automatically generated



The results:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

## 4.Run P4 version of Airport simulation

Text

Description automatically generated

The results:

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

Text

Description automatically generated

## 5.Run P5 version of Airport simulation

Text

Description automatically generated

The result:

Text

Description automatically generated

Graphical user interface, text

Description automatically generated

# Conclusion

1. There are 24 files in my project.

Table

Description automatically generated

1. The main idea:

Diagram

Description automatically generated

Main\_1: P1 version of Airport simulation

Main\_2: P2 version of Airport simulation

Main\_3: P3 version of Airport simulation

Main\_4: P4 version of Airport simulation

Main\_5: P5 version of Airport simulation

Diagram

Description automatically generated

Box and whisker chart

Description automatically generated

Random is the file that imported into each main files.

Diagram

Description automatically generated

Specific idea for each simualation:

#### Main\_1

A picture containing chart

Description automatically generated

#### Main\_2

Set up two runways in main\_2

One runway is used for landings, one for departures.

One plane can land or depart in each unit of time.

If the departure runway can’t implement the can\_depart method, the plane will be refused.

If the arrival runway can’t implement the can\_land method, the plane will be refused.

#### Main\_3

One runway is normally used for landings, the other is normally for departures.

One plane can land or depart on each runway in each unit of time.

If the size of the departure runway and arrival runway is 0, the plane can land on the arrival runway, or the plane can depart from the departure runway.

If arrival and departure runways can implement the can\_land function, the plane will be refused.

#### Main\_4

One runway is used for landings, a second is for departures.

The third runway (Runway overflow) is for overflow traffic. It gives priority to arrivals. One plane can land or depart on each runway in each unit of time.

Based on the state of two runways: departure and arrival runways, we will choose the suitable method for the third runway.

#### Main\_5

Add Plane\_p5.h Plane\_status emergency and variable fuel.

The fuel will determine by random.h file

If fuel< 1, assign the emergency situation to state variable.

Edit in the main\_5 :If the state is emergency, we will require the plane can land immediately.