**Table of Content**

1. Introduction to overall report and task

2. Objective of this Task

3. Analysis

3.1 Analysis as a Solution

3.1.1 Basic

3.1.2 Intermediate

3.1.3 Advance

4. Group discussion and work

5. Design

5.1 Flowchart

5.2 Algorithm

6. Implementation

7. Testing

7.1 Walk through

8. Conclusion

9. References

**Introduction :**

This is the second assessment provided by the Northampton University for the psp module (CSY1020) to the Computing Students of year 1 . The report reflects about the scenario as provided, where a Golden Ball is an actor moving around the maze as suggested in the proper direction using left, right, up and down key in the 16x13 grid of maze with one movement per block of sand. Three rules to be followed during this assessment is to complete the basic, intermediate and advance level using the idea of looping, conditional statement, constructors, method, parameters and many other parts of java using eclipse making a new GUI. Ideas for this assessment is generated from group as well as individual work . Several processes involved during this report can be highlighted as starting from the group discussion for the design interface , knowing task briefly and giving it a solution using analysis as a main backbone ,design for the lookup, implementation for the ideas generated and testing, in order to know the faults and the limitations of this application.

**Objective of this task:**

Meeting the whole criteria like basic, intermediate and advance is the objective of this task. In basic a Golden Ball should move only one step per one click in the proper direction as up, down, right and left using ^, v, >, < . Not only this, on pressed of run button, the ball should move automatically forward and stops when it is in the white sandstone. Should move with act button also a move at a time . Compass button should show the direction of ball in link with the direction field, likewise the position of the ball should be seen in the text field of square and the option buttons on press of which scenario changes and the pressed button should be seen in TextField of option. Regarding intermediate the ball should fall automatically down if there is a sand below it without dropping into white spaces and a sound should be play on fall. Beside this, in the advance level using extra ideas should be implemented as per the good design.

**Analysis :**

This is the main phase of this assessment where the solution for the application is carried out and then coded using java in eclipse and rewrite if error’s available. Before the design phase, this is the fundamental core through which every rules meets as required and presented . The discussed and every requirement are analyzed and generates in this section in the theoritical form which is later translated in codes. New GUI is to be created and then the program should run in that with the help of theoritical solution changing to the code format. For the function of this program buttons like act, reset, run, keys, options are to be used.

**Analysis as a Solution :**

**1.Basic rule :**

In this phase, conditional statement is used to move ball after creating a maze and giving the ball condition to check the sand is equals to next block or not followed by array with the variables i and j as row and column. Whenever, there is white block, the conditional statement and looping makes the ball to be restricted using the same way as for the movement of ball. Move ball method is also used so that ball can move using ^, v, < and > buttons as required by the scenario using action listener for the function. As well as movement of a ball automatically and one step forward at a time with the pres of run and act button is implemented using conditional statement.

**2.Intermediate :**

Falling of ball automatically whenever there’s a sand block below was implemented using timer and giving a interval to run the function and calling it method moveDown, which helps the ball to fall instantly in the given interval of time.

**3.Advance :**

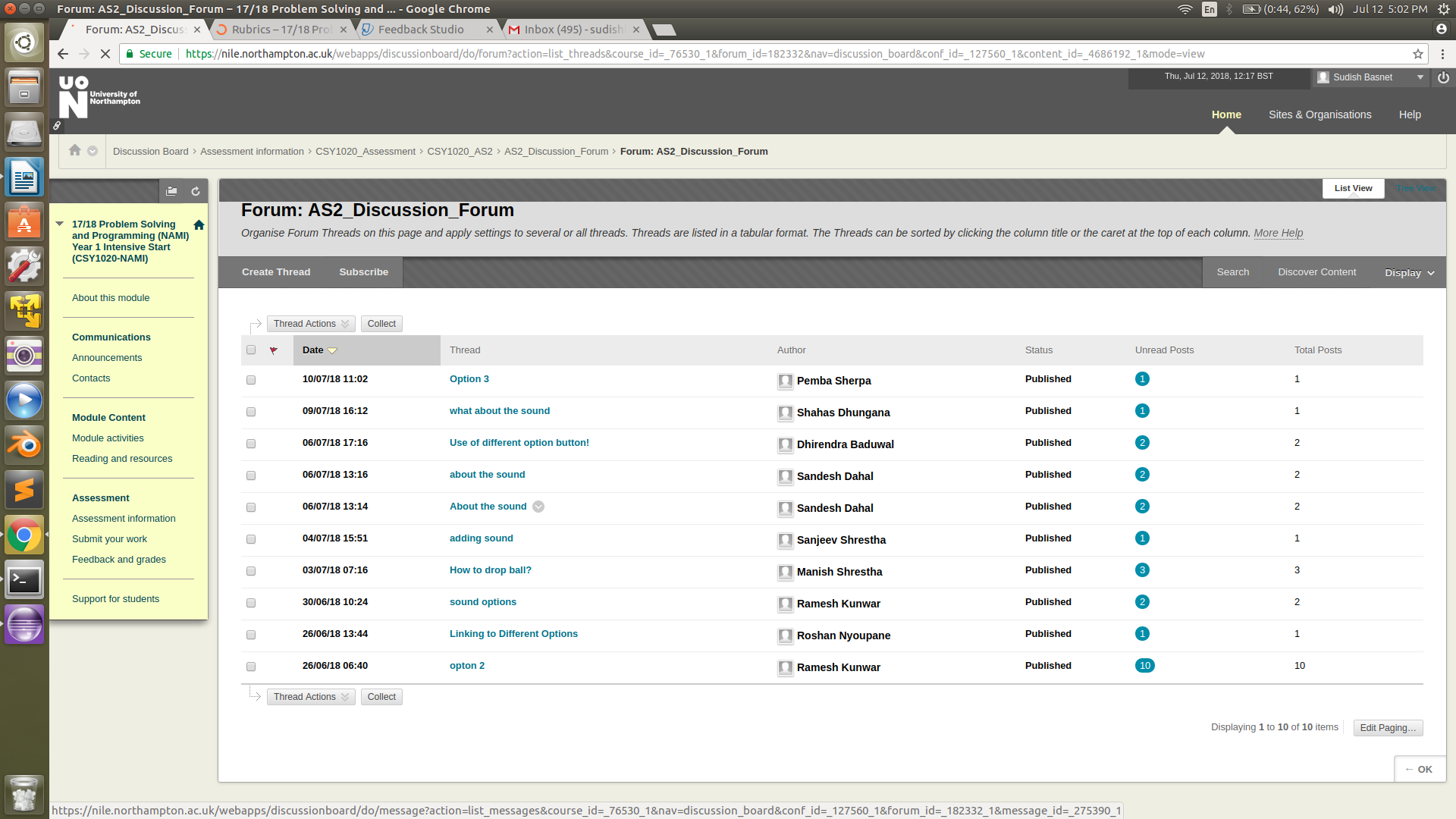
For the advance level, as per the discussion of group meeting, skull as an obstacle and strawberry as a score increaser is used and introduced using ImageIcon constructor and giving it as certain position using array and putting it by the help of conditional statement and looping.

**Group discussion and work :**

For the design phase, group discussion was done to implements new ideas and merging in a single idea. Main focus of this group discussion was for the advance phase in this assessment to introduce new actor acting as a food and another as a obstacle where the idea for the obstacle is to make them move randomly. Beside that after all, idea for this obstacle was finalize with the method of making only a obstacle move and rest of the other to be constant as per the requirement of new players regarding difficulty level.

Many ideas were discussed which is the benefits of this group discussion so that the application was implemented as good as thought. Design was successfully done and the all requirements meets.

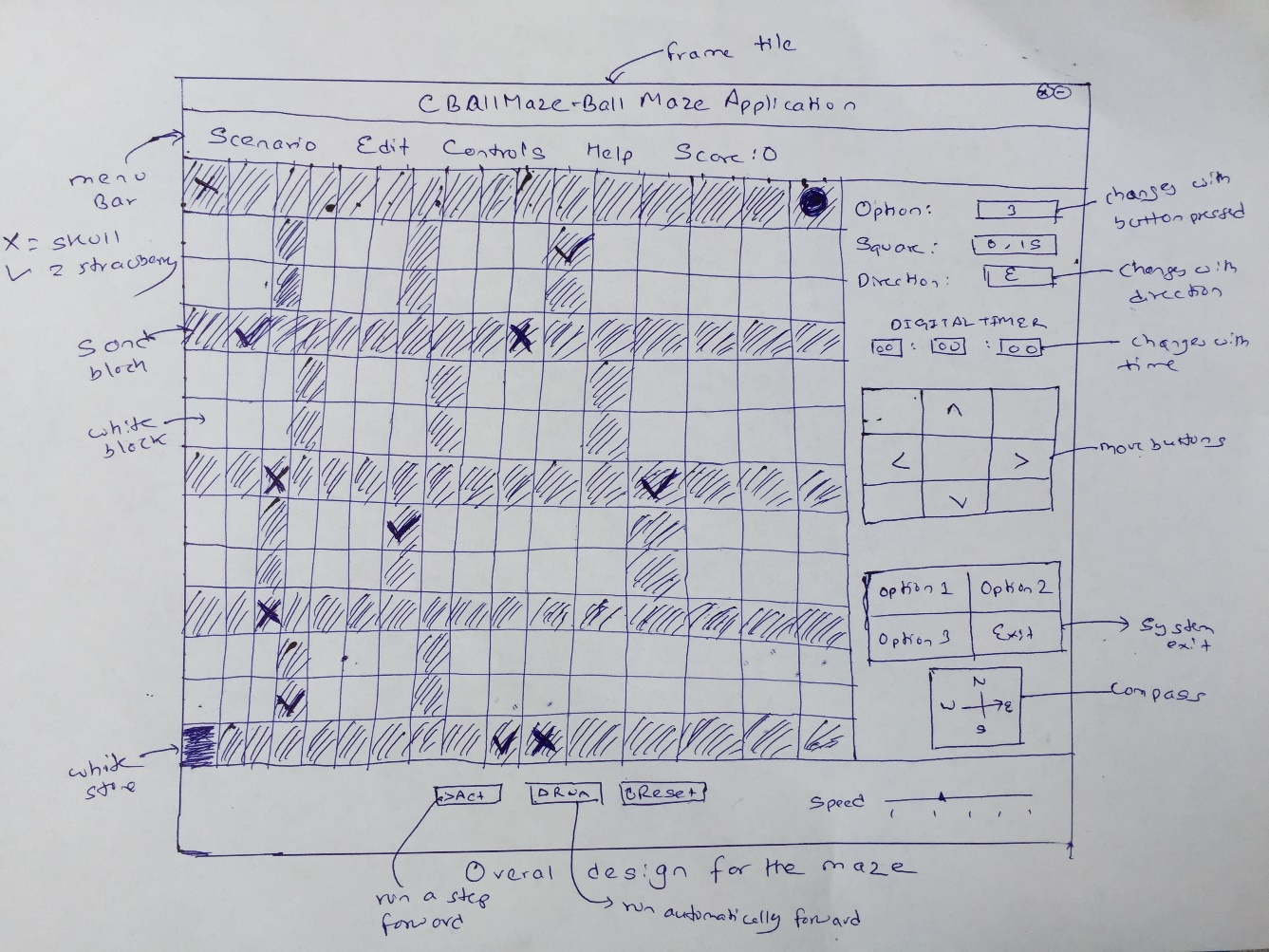
Beside this, due to same ideas there was some problem regarding code to be matched. It was also difficult to choose the idea for the implementation for finalizing the application.



**Fig:1.1**

**Design :**

In this phase the overall layout and the process involved with all the function of the application is discussed with an algorithm and flowchart, so the whole program can be examined.



**Flowchart:**

Can ball move ?

Restart

No

yes

Move

Is enemy touched?

yes

Game Over

Is there box below?

No

Free Fall

Move

No

Is Ball touches Strawberry?

Is ball in the white stone?

yes

Increase point in counter

Mission Complete

**Fig 1:** Flowchart of the whole system

Run

Press key left, right, up or down

No

If pressed up

If pressed down

If pressed right

If pressed left

No

No

No

Yes

If can move

No OO

No

Yes

If can move

If can move

Yes

If can move

Yes

No

No

Yes

Yes

Yes

Yes

Move up(per 1 block sand)

Move right(per 1 block sand)

Move Left(per 1 block sand)

)

Move down(per 1 block sand)

**Fig** 1.1: Basic Flowchart

Run

Move left, right

Is there a sand box below?

Yes

No

Ball falls automatically

With a sound effect

Is no sand box below ball ?

Yes

No

**Fig 1.2:** Intermediate flowchart

**Algorithm:**

**Algorithm for Basic :**

1. Algorithm for run including option, square, direction :

Step 1: Start

Step 2: Select the option

Step 3: Press the run button

Step 4: Ball move automatically forward

Step 5: Square=position of ball, Option = Option press, Direction = direction of ball

Step 6: Ball stops at white stone

Step 7: Stop

2. Algorithm for act :

Step 1: Start

Step 2: Press the act button

Step 3: Move one step forward

Step 4: Press the move key

Step 5: Move with change of one block of sand

Step 6: Ball stops at white stone

Step 7: Message Dialog “Welldone”

Step 8: Stop

**Algorithm for Intermediate:**

Step 1: Start

Step 2: Move ball

Step 3: Check for the sand block below ball

Step 4: If sand box below=True

Step 5: Fall automatically in the lowest sand block, Play sound

Step 6: Stop

**Algorithm for Advance**

Step 1: Start

Step 2: Move ball

Step 3: If Skull=True, Game over

Step 4: If Strawberry=True, Counter=Point+1

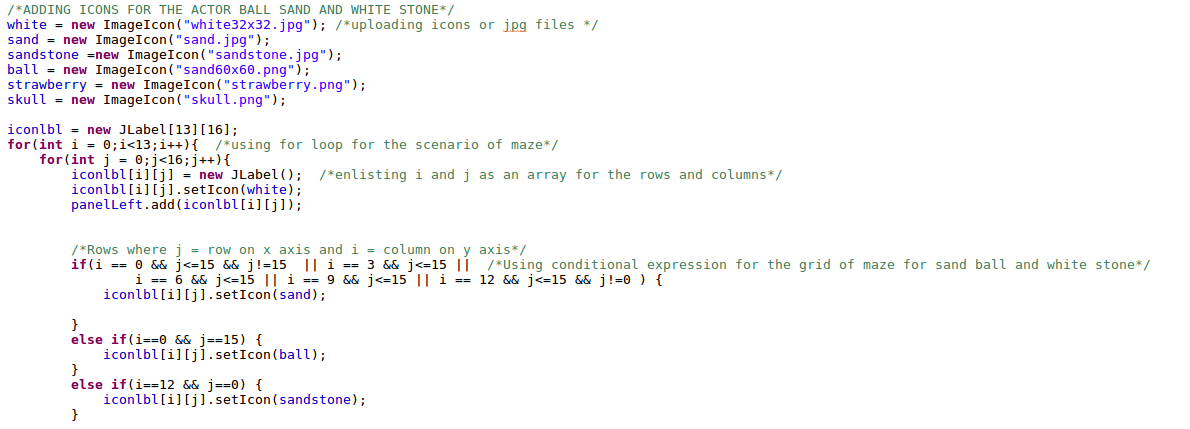
Step 5: If ball is last silver block, Mission=Success

Step 6: Stop

**Implementation :**

Different implementation had been carried out in order to arrange the code in correct form and giving the proper function. The main implementations carried out are as creating maze, strawberry, skull, ball, digital timer, menu, general textfields and labels. Commentary of the code explaining design applied and Screen shots of snippets of the code and the outcome in the GUI are shown below.

1. Making maze and adding jpg or png files in the scenario with the help of array, variables and nested loops as well as conditional statement to create a maze and adding actors as shown in the below figure:



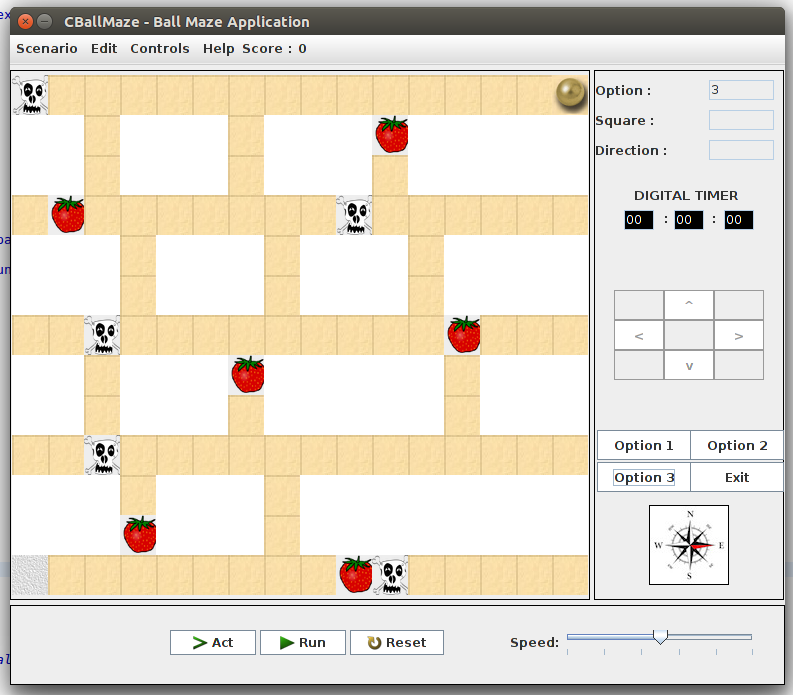


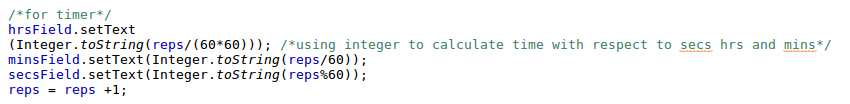
fig:1.2

fig : 1.1

2. Making a new method in action listener for the key movement called moveLeft and giving it condition to move. Here a button is created using JButton and then action listener is introduced as new and called a new method. Using conditional statement movement of ball is enabled with the help of array for the row and column. The implementation for this movement can be seen in video of testing.



3. Digital timer : After creating three text fields and 2 Jlabel for “:”, in the action listener of the hrsField, minsField, secsField the following code is implemented. Before this an initial value for reps is already given in the main method which is set to 0.



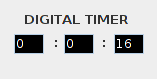
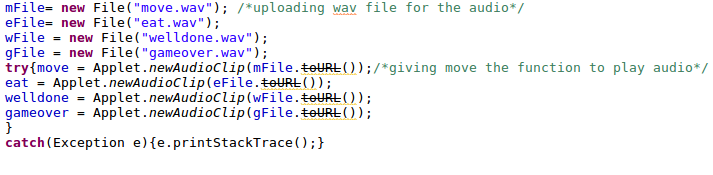


fig:1.3

4. Implementing audio file in order to play during some function using try catch expression where the .wav file is uploaded using File constructor and call the file name in try for applet to new audio clip. Where ever the audio is required, using audioname.play ()starts the audio.



5. Implementing message dialog box: Using the code >> JOptionPane.*showMessageDialog*(**null**,"GAME OVER !!"); the message dialog box is seen in a new frame, where the message GAME OVER can be replaced with other messages too. Sample of this dialog box is below, which is the output of the submenu about:

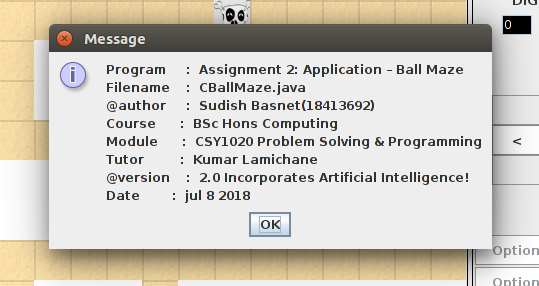


fig : 1.4

**Testing:**

This is the most important phase in this assignment. Further completeness of the application is totally dependent on testing. Not only which function works but also the unexpected errors and their outcomes for the improvement are involved here.

Starting with the functions that works perfect with the inputs are set below in the walk through:

|  |  |  |
| --- | --- | --- |
| **Set of input** | **Expected output** | **Actual output** |
| Run key clicked | Run automatically forward | Run |
| Reset key clicked | dispose() | Scenario reset |
| Act key clicked | Move a step forward | Move one time |
| Left key clicked | Move left covering one sand block | Left moved |
| Right key clicked | Move right covering one sand block | Right moved |
| Down key clicked | Move down covering one sand block | Down moved |
| Up key clicked | Move up covering one sand block | Up moved |
| Option button clicked | Check the frame in link with option | frame |
| Eat strawberry | Score = a++ | Increase point on score board |
| Touch Skull | Get object out of set | Game over |
| Touch the final silver box | Game Stop | Well Done Mission Complete |
| Exit | System.exit(0) | Frame closes |

Table 1.1

**Conclusion**

**References**

1. Java2s.com. (2018). *Applet: newAudioClip(URL audioFileURL) : Applet « java.applet « Java by API*. [online] Available at: http://www.java2s.com/Code/JavaAPI/java.applet/AppletnewAudioClipURLaudioFileURL.htm [Accessed 12 Jul. 2018].

2. www.javatpoint.com. (2018). *Java for Loop - Javatpoint*. [online] Available at: https://www.javatpoint.com/java-for-loop [Accessed 12 Jul. 2018].

3. game, S. (2018). *Score sytem for my java game*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/21476923/score-sytem-for-my-java-game [Accessed 12 Jul. 2018].