|  |  |  |
| --- | --- | --- |
| *../../../../Desktop/Screen%20Shot%202017-02-06%20at%2013.23.* | **Name:…………………………….**  **Student ID:……………………..**  **Comments (tutor)…………………..……….**  **……………………………..………**  **……………….……………………**  **……………………………………**  **……………………………………** | |
| System Requirements: Essential (Graphical User Interface): | |  |
| 13 x 16 grid of **JButton**’s or Icon’s. | |  |
| 4 **JButton**’s for the game options ‘2 Player, 4 Player, Multi’ and ‘*Exit*’. | |  |
| 3 **JButton**’s for ‘Act’, ‘Run’ and ‘*Reset*’. | |  |
| 9 **JButton**’s for ‘*Forward >*’, ‘Backwards <’, ‘Up ^’, ‘Down v’ should move the ball in the appropriate direction by one square for each press (plus 5 blank). | |  |
| The compass icon (**JButton)** should illustrate the current direction for the ball. | |  |
| **3 JLabel**’s for ’Option’, ‘*Square*’ and ‘*Direction*’. | |  |
| **3 JTextField**’s for the current ‘Option’, Location/*’Square’* and *‘Direction’* of the ball. Use the square identification method e.g. 0 to 207 and N, E etc. | |  |
| 3 **JLabel’s** for the ‘DIGITAL TIMER and the two :’, with 3 **JTextField’**s for the hours, minutes and seconds. | |  |
| 2 **JLabel’s** for the ‘SCORE and ‘<L:R>’, with 2 **JTextField’**s for the scores (L & R). | |  |
| Create a **JFrame** application, which opens to the set size (825 \* 585). | |  |
| **JFrame** title set as "*CBabyBallBounce – Baby Ball Bounce Application*". | |  |
| System Requirements: Additional (Functionality & Complexity): | |  |
| Application icon for the **JFrame** used. | |  |
| The ‘Run’ **JButton** should show the ball moving between the babies continuously from the initial position (2 Player – default opening state). | |  |
| The ‘Reset’ **JButton** should clear/reset the application to its starting/default opening state. | |  |
| The ‘Act’ **JButton** should step through the above ‘Run’ sequence one move at a time. | |  |
| Discuss and implement the different options for the 3 configurations. | |  |
| The ‘2 Player, 4 Player, Multi’ **JButton**’s should display different obstacle/car configurations/locations. | |  |
| A **JMenuBar** could be included with **JMenu**’s for the *Scenario, Edit, Controls* and *Help*, which include **JMenuItem**’s of *Exit (Scenario)*, *Help Topic* and *About (Help)*. | |  |
| Additional **JButton**’s may be used to improve the applications usability e.g. ball bounce – in random direction, deflection angle etc. | |  |
| Create a **JFrame** application, which is not resizable. | |  |
| Create a **JFrame** application, which centres itself on the monitor. | |  |
| Use of additional baby images indicating the current position and direction of the baby. | |  |
| Discuss the possibilities for incorporating intelligence/checks for whether moves are valid. | |  |
| Digital Timer should start and stop when run is pressed and stopped when a baby misses the ball (with the ball continuing to the left or right boundary and stopping itself and the timer). | |  |
| Implement intelligence/checks for whether moves are valid. | |  |
| A **kickBall()** method should be used to solve the problem. The **kickBall()** method should include **move(left), move(right), move(up), move(down)** methods | |  |
| **CBabyBallBounce.java** & **CBabyBallBounce.class** | |  |

Key: Blue GUI; Yellow Testing Application; Red Code.