**DETAILED ALGORITHM (15 points)**

1. Welcome user to the program and explain game rules.
2. Show menu options to the user.
3. Ask the user to select an option. (*If user wants to play – option P. If user wants to quit game – option Q*).
4. Set entered information to the upper case.
5. If user entered ‘*P*’ then generate a random username and welcome the user to the sub menu (then go to 6). If user entered ‘Q’ display message “Thank you for using this program” to the user and terminate the program.
6. Show sub menu option to the user (*If user wants to start flipping coins – option F. If user wants to quit to the main menu – option Q*)
7. If user decided to flip coins (entered ‘*F*’) – start the process (go to 8). If user decided to quit to the main menu (entered ‘*Q*’), start at 3 again.
8. Flip all four coins. If coin lands heads up add the value of the coin to the balance. If coin lands tail up nothing will be added to the balance.
9. Display current balance to the user.
10. Display message to the user either he/she won or lost or can continue playing.
    * 1. If balance is equal $1.00 user wins the game (go to 11).
      2. If balance is over $1.00 user losses the game (go to 11).
      3. If balance is less than $1.00 user can continue playing (go to 7).
11. Ask the user if she/he wants to play the program again (*Y – play again, N – go to main menu*).
    1. If user entered ‘*Y*’ start at 6 again.
    2. If user entered ‘*N*’ start at 3 again.

**UML DIAGRAMS (15 points)**

|  |  |  |
| --- | --- | --- |
| + | Main | |
| + | welcome() | void |
| + | main(String[]) | void |

|  |  |  |
| --- | --- | --- |
| + | Menu | |
| - | menuOption | String |
| - | subMenuOption | String |
| - | keyboard | Scanner |
| - | playerNames | String[] |
|  |  |  |
|  | Menu() | constructor |
|  |  |  |
| + | getScanner() | Scanner |
| + | setMenuOption(String) | void |
| + | getMenuOption() | String |
| + | setSubMenuOption(String) | void |
| + | getSubMenuOption() | String |
| + | thankyou() | void |
| + | menuOptions() | void |
| + | subMenuOption() | void |

|  |  |  |
| --- | --- | --- |
| + | Player | |
| - | penny | Penny |
| - | nickel | Nickel |
| - | dime | Dime |
| - | quarter | Quarter |
| - | balance | int |
| - | gameOver | boolean |
| - | playerName | String |
|  |  |  |
|  | Player(String) | constructor |
|  |  |  |
| + | setPlayerName(String) | void |
| + | getPlayerName() | String |
| + | setBalance(int) | void |
| + | getBalance() | int |
| + | setGameStatus(boolean) | boolean |
| + | flipAgain() | boolean |
| + | start() | void |

|  |  |  |
| --- | --- | --- |
| + | Penny | |
| - | value | int |
| - | coin | Coin |
|  |  |  |
|  | Penny() | constructor |
|  |  |  |
| + | getValue() | int |
| + | setValie(int) | void |
| + | coinFlip() | int |

|  |  |  |
| --- | --- | --- |
| + | Dime | |
| - | value | int |
| - | coin | Coin |
|  |  |  |
|  | Dime() | constructor |
|  |  |  |
| + | getValue() | int |
| + | setValie(int) | void |
| + | coinFlip() | int |

|  |  |  |
| --- | --- | --- |
| + | Nickel | |
| - | value | int |
| - | coin | Coin |
|  |  |  |
|  | Nickel() | constructor |
|  |  |  |
| + | getValue() | int |
| + | setValie(int) | void |
| + | coinFlip() | int |

|  |  |  |
| --- | --- | --- |
| + | Quarter | |
| - | value | int |
| - | coin | Coin |
|  |  |  |
|  | Quaeter() | constructor |
|  |  |  |
| + | getValue() | int |
| + | setValie(int) | void |
| + | coinFlip() | int |

**VARIABLES/IPO (5 points)**

value(int)

menuOption(String)

subMenuOption(String)

balance(int)  
gameOver(Boolean)

playerName(String)

**FORMULAS (5 points)**

balance/100

balance+penny(amount)

balance+nickel(amount)

balance+dime(amount)

balance+quarter(amount)

**JAVA CODE - submit the entire Java folder in compressed (zipped) format (50 points)**

Submitted separately.