|  |  |  |
| --- | --- | --- |
| **INPUT** | **PROCESS** | **OUTPUT** |
| ask **rounds**  ask **vowelsAmnt**  ask **consonantsAmnt**  ask **p2Answ**  ask **p1Answ**  **playGameAgain** | **computeRandomLetters** – Firstly loops the length of **vowelsAmnt** and **consonantsAmnt** and progresses by 1 so that the user can input in how many of each they would like.  Runs calculations for generating random vowels and consonants **(Math.random** on the arrays **VOWELS** and **CONSONANTS)**, then sorts the letters into a String called **randomLetters** using a String Buffer.  **strBuff.append** stores **mixedVowels** and **mixedConsonants** in the **StringBuffer**.  **randomLetters=strBuff.toString();**  **computeScore** – Runs calculations to see if the word inputted by the player is present in the word bank or not, also compares the length of each players inputted answer and as a result, rewards the player with the longest, correct word with a point.  If **p1Answ.equals(wordBank[i])**  **p1Answ.length()>p2Answ.length()||p1Answ.length()==p2Answ.length()**  **p1Score = p1Score + 1;**  If **p2Answ.equals(wordBank[i])**  **p2Answ.length()>p1Answ.length()||p2Answ.length()==p1Answ.length()**  **p2Score = p2Score + 1;**  **computeWinner** – Runs calculations for different outcomes of scoring to see who wins the game, also accounts for a draw.  If **p1Score>p2Score**  if **p2Score>p1Score**  else Draw  **computeReset** – This initialises/resets each player’s scores to zero when the game loops/repeats so that the scores don’t keep getting added to the total.  **P1Score**=0;  **P2Score**=0;  If **playGameAgain** = True (Y), then game will loop/repeat. If False (N), the game will end. | display **p1Score**  display **p2Score**  reveal **winnerAnnouncement** |