**PSEUDOCODE**

**MODULE 1: (Answer Class)**

Begin.

Declare input variables.

End.

**MODULE 2: (FindingDifficultyClass)**

Begin

Get the input details as file.

If(QuestionType is MCQ)

Call setValues() and set input values in MCQClass.

Call findDifficulty().

Else if(QuestionType is Fillups)

Call setValues() and set input values in FillUpsClass.

Call findDifficulty().

Else if(QuestionType is Match)

Call setValues() and set input values in MatchClass.

Call findDifficulty().

Else if(QuestionType is Programming)

Call setValues() and set input values in ProgramClass.

Call findDifficulty().

Else

Print “Invalid Question Type”.

Print questionType, levelOfDifficulty, rightAnswer percentage, wrongAnswer percentage, languageMostlyUsed and maxMarks.

If input file is invalid

Print “Invalid Input File”.

End

**MODULE 3 (MCQClass)**

Begin.

Declare input variables.

Assign input variables in setValues() function.

**findDifficulty() function :**

Find rightAnswerScore = students\_AnsweredRight / studentsAttented.

Find wrongAnswerScore = students\_AnsweredWrong / studentsAttented.

Assign difficultyIndex = rightAnswerScore

Calculate the number of students given Hard, Medium and Easy as feedback.

Modify difficultyIndex.

If(difficultyIndex < 0.4)

levelOfDifficulty is HARD.

Else if(difficultyIndex >= 0.4 and difficultyIndex <=0.6)

levelOfDifficulty is MEDIUM.

Else

levelOfDifficulty is EASY.

Return answer as type Answer.

End.

**MODULE 4 (FillUpsClass)**

Begin.

Declare input variables.

Assign input variables in setValues() function.

**findDifficulty() function :**

Find rightAnswerScore = students\_AnsweredRight / studentsAttented.

Find wrongAnswerScore = students\_AnsweredWrong / studentsAttented.

Assign difficultyIndex = rightAnswerScore

Calculate the number of students given Hard, Medium and Easy as feedback.

Modify difficultyIndex.

If(difficultyIndex < 0.4)

levelOfDifficulty is HARD.

Else if(difficultyIndex >= 0.4 and difficultyIndex <=0.6)

levelOfDifficulty is MEDIUM.

Else

levelOfDifficulty is EASY.

Return answer as type Answer.

End.

**MODULE 5 (MatchClass)**

Begin.

Declare input variables.

Assign input variables in setValues() function.

**findDifficulty() function :**

Find rightAnswerScore = students\_AnsweredRight / studentsAttented.

Find wrongAnswerScore = students\_AnsweredWrong / studentsAttented.

Find partiallyRightAnswerScore = students\_AnsweredPartiallyRight / 2\*studentsAttented.

Assign difficultyIndex = rightAnswerScore + partiallyRightAnswerScore.

Calculate the number of students given Hard, Medium and Easy as feedback.

Modify difficultyIndex.

If(difficultyIndex < 0.4)

levelOfDifficulty is HARD.

Else if(difficultyIndex >= 0.4 and difficultyIndex <=0.6)

levelOfDifficulty is MEDIUM.

Else

levelOfDifficulty is EASY.

Return answer as type Answer.

End.

**MODULE 6 (ProgramClass)**

Begin.

Declare input variables.

Assign input variables in setValues() function.

**findDifficulty() function :**

Find rightAnswerScore = students\_AnsweredRight / studentsAttented.

Find wrongAnswerScore = students\_AnsweredWrong / studentsAttented.

Find partiallyRightAnswerScore = students\_AnsweredPartiallyRight / 2\*studentsAttented.

Find mostlyUsedLang.

Assign difficultyIndex = rightAnswerScore + partiallyRightAnswerScore.

Calculate the number of students given Hard, Medium and Easy as feedback.

Modify difficultyIndex.

If(difficultyIndex < 0.4)

levelOfDifficulty is HARD.

Else if(difficultyIndex >= 0.4 and difficultyIndex <=0.6)

levelOfDifficulty is MEDIUM.

Else

levelOfDifficulty is EASY.

Return answer as type Answer.

End