**Calculating Exam Statistics Program**

**Project Summary**

This program is for calculating exam statistics: number of grades, average grade, maximum grade, minimum grade, number of grades above the average, and number of grades below the average.

**Getting Started**

Download Visual C++ 2017 Community: A free Windows C++ compiler by Microsoft.

Download DEV-C++ 5.11

**How to run the program**

Step 1: Configure Dev-C++.

We need to modify one of the default settings to allow you to use the debugger with your programs.

Step 2: Create a new project.

A “project” can be considered as a container that is used to store all the elements that are required to compile a program.

Step 3: Create/add source file(s).

Step 4: Compile.

Once you have entered all of your source code, you are ready to compile.

Step 5: Execute.

You can now run your program.

Step 6: Debug.

When things aren’t happening the way you planned, a source-level debugger can be a great tool in determining what really is going on. Dev-C++’s basic debugger functions are controlled via the “Debug” tab at the bottom of the screen; more advanced functions are available in the “Debug” menu.

**Instruction of the program**

**C++ standard library headers**

|  |  |
| --- | --- |
| **Library headers** | **Description** |
| #include <iostream> | Header that defines the standard input/output stream objects. |
| #include <fstream> | Header that providing file stream classes. |
| #include <cmath> | Header that declares a set of functions to compute common mathematical operations and transformations. |
| #include<iomanip> | Header providing parametric manipulators. |

**using namespace std;**

are used to define a scope and allows us to group global classes, functions and/or objects into one group.

**Prototype Functions**

|  |  |
| --- | --- |
| **Prototype Functions** | **Description** |
| int numberOfGrade (string fileName); | It is a prototype function that calculate the number of grades and return value as an integer from the string fileName. |
| float calculateAverageGrade (string fileName); | It is a prototype function that calculate the average grade and return value as a float from the string fileName. |
| float calculateHighestGrade (string fileName); | It is a prototype function that calculate the highest grade and return value as a float from the string fileName. |
| float calculateLowestGrade (string fileName); | It is a prototype function that calculate the lowest grade and return value as a float from the string fileName. |
| int calculateAboveAverage (string fileName); | It is a prototype function that calculate grades that above average and return value as an integer from the string fileName. |
| int calculateBelowAverage (string fileName); | It is a prototype function that calculate grades that below average and return value as an integer from the string fileName. |

**int main() function** - This line initiates the declaration of a function.

**In the main function:**

Declare string fileName = “grade.txt”;

|  |  |
| --- | --- |
| **Print out** | **Descriptions** |
| cout << “The number of grades is “<< numberOfGrade(fileName) << endl; | Print out the sentence that indicate number of grades and result from the function numberOfGrade(fileName). |
| cout << “The average grade is “<< calculateAverageGrade(fileName) << endl; | Print out the sentence that indicate the average grade and result from the function calculateAverageGrade(fileName). |
| cout << “The highest grade is” << calculateHighestGrade(fileName) << endl; | Print out the sentence that indicate the highest grade and result from the function calculateHighestGrade(fileName). |
| cout << “The lowest grade is” << calculateLowestGrade(fileName) << endl; | Print out the sentence that indicate the lowest grade and result from the function calculateLowestGrade(fileName). |
| cout << “The number of grades above the average is” << calculateAboveAverage(fileName) << endl; | Print out the sentence that indicate the grade above the average and result from the function calculateAboveAverage(fileName). |
| cout << “The number of grades below the average is “<< calculateBelowAverage(fileName) << endl; | Print out the sentence that indicate the grade below the average and result from the function calculateBelowAverage(fileName). |

At the end of the main function, we need to write return 0, to return all the values and It is like saying the program works fine.

**Outside of main functions, working on six functions individually:**

|  |  |
| --- | --- |
| 1. **int numberOfGrade (string fileName)** | |
| Open the file | ifstream inFile;  inFile.open(“grade.txt”); |
| Declare variables for this function | int counter = 0;  //counter starts from 0. |
| Using while loop for inFile | while (inFile)  {  float grade;  //declare grade as a float  inFile >> grade;  //pull grades from the inFile  If (inFile)  {  counter++;  }  }  //using if, count all the grades that exist in the inFile. |
| Return counter after the while loop | |

|  |  |
| --- | --- |
| 1. **float calculateAverageGrade (string fileName)** | |
| Open the file | ifstream inFile;  inFile.open(“grade.txt”) |
| Declare variables for this function | float total = 0;  float counter = 0; |
| Using while loop for inFile | while (inFile)  {  float grade;  //declare grade as a float  inFile >> grade;  // pull all grade from the inFile  If(inFile)  {  Total += grade;  counter++;  }  }  //using if, adding the points that have in each grade that exist in the inFile together, to the variable calls Total. |
| Outside the while loop | Total = total /counter;  Return total;  // after finding the total grade, divide it by counter, which is the number of grades. Return total for final result. |

|  |  |
| --- | --- |
| 1. **float calculateHighestGrade (string fileName)** | |
| Open the file | ifstream inFile;  inFile.open(“grade.txt”); |
| Declare variables for this function | float highest = 0;  Int counter = 0; |
| Using while loop for inFile | while(inFile)  {  float grade;  //declare grade as a float  inFile >> grade;  //pull all grade from the inFile  If(inFile)  {  counter++;  If (grade > highest)  {  Highest = grade;  }  }  }  //make grade is bigger than highest grade and count all the highest grade. And then declare highest grade =grade. |
| Outside of while loop | return highest;  //To get the final result |

|  |  |
| --- | --- |
| 1. **float calculateLowestGrade (string fileName)** | |
| Open the file | ifstream inFile;  inFile.open(“grade.txt”); |
| Declare variables for this function | float lowest= 0;  int counter = 0; |
| Using while loop for inFile | while(inFile)  {  float grade;  //declare grade as a float  inFile >> grade;  //pull all grade from the inFile  If(inFile)  {  counter++;  If (grade <lowest)  {  lowest = grade;  }  }  }  //make grade less than lowest grade and count all the lowest grade. And then declare lowest grade =grade. |
| Outside of while loop | return lowest;  //To get the final result |

|  |  |
| --- | --- |
| 1. **int calculateAboveAverage (string fileName)** | |
| Open the file | ifstream inFile;  inFile.open(“grade.txt”); |
| Declare variables for this function | Float average = calculateAverageGrade(fileName);  int counter = 0; |
| Using while loop for inFile | while(inFile)  {  float grade;  //declare grade as a float  inFile >> grade;  //pull all grade from the inFile  If(inFile)  {  If (grade > average)  {  Counter++;  }  }  }  //count all the grades that higher than average. |
| Outside of while loop | return counter;  //To get the final result |

|  |  |
| --- | --- |
| 1. **int calculateBelowAverage (string fileName)** | |
| Open the file | ifstream inFile;  inFile.open(“grade.txt”); |
| Declare variables for this function | Float average = calculateAverageGrade(fileName);  int counter = 0; |
| Using while loop for inFile | while(inFile)  {  float grade;  //declare grade as a float  inFile >> grade;  //pull all grade from the inFile  If(inFile)  {  If (grade < average)  {  Counter++;  }  }  }  //count all the grades that less than average. |
| Outside of while loop | return counter;  //To get the final result |