# CSC242 Introduction to Programming Concepts Week Three Assignment

| **First Name** | David |
| --- | --- |
| **Last Name** | Leon |
| **ID#** | 041224771 |
| **Email Address** | d.leon771@student.nu.edu |

Table of Contents

[How to submit your Assignment 2](#_Toc535167619)

[P1 – Temperature Conversions 2](#_Toc535167620)

[P2 – Displaying a Square of Asterisks 3](#_Toc535167621)

[P3 – Perfect Numbers 3](#_Toc535167622)

[P4 – Assign Grades 4](#_Toc535167623)

[P5 – Pattern Recognition: Consecutive Four Equal Numbers 4](#_Toc535167624)

[P6 – Common Elements 5](#_Toc535167625)

# 

# How to submit your Assignment

After filling all the parts in this file, please follow the following steps.

1. Add your name and ID to the first page.
2. Save the file in the original format (Docx or Doc)

(please do not convert to other file formats e.g. PDF, ZIP, RAR, …).

1. Rename the file as

CSC242 *–* ***HW3*** *- ID – YOUR Last Name - YOUR First Name.docx*

**Example:** CSC242 *–* ***HW3*** *-* 234566435 - Smith - John.docx

1. Upload the file and submit it (only using Blackboard)

# P1 – Temperature Conversions

Implement the following integer functions:

**a)** Function ***celsius*** returns the Celsius equivalent of a Fahrenheit temperature, using the calculation

*celsius = 5.0 / 9.0 \* (fahrenheit - 32);*

**b)** Function ***fahrenheit*** returns the Fahrenheit equivalent of a Celsius temperature, using the calculation

*fahrenheit = 9.0 / 5.0 \* celsius + 32;*

|  |
| --- |
| Your C++ code for this problem |
| #include <iostream>  #include <string>  using std::cout;  using std::endl;  using std::cin;  double celcius( const int f)  {  return ( 5.0 / 9.0 ) \* (f - 32) ;  }  double fahrenheit( const int c)  {  return ( 32 + ( c \* 9.0 / 5.0) );  }  int main()  {  int fahr = 102.0;  int cel = 38.888;  double ans\_f = celcius(fahr);  double ans\_c = fahrenheit(cel);  return 0;  } |

Run the code and insert the result in the following box.

|  |
| --- |
| The run result |
| A screenshot of a computer screen  Description automatically generated |

# 

# P2 – Displaying a Square of Asterisks

Write a function **squareOfAsterisks** that displays a solid square (the same number of rows and columns) of asterisks whose side is specified in integer parameter **side**. For example, if **side** is 4, the method should display

|  |
| --- |
| \*\*\*\*  \*\*\*\*  \*\*\*\*  \*\*\*\* |

In your **main()** function, read an integer value function for side from the user and call the **squareOfAsterisks** and display the asterisks.

|  |
| --- |
| Your C++ code for this problem |
| #include <iostream>  #include <string>  using std::cout;  using std::endl;  using std::cin;  void squareOfAsterisks(unsigned n)  {  for (size\_t row = 0; row < n; row++)  {  for (size\_t col = 0; col < n; col++)  {  cout << "\*";  }  cout << endl;  }  }  int main()  {  unsigned n = 0;  cout << "Enter num ";  cin >> n;  squareOfAsterisks(n);  return 0;  } |

Run the code and insert the result in the following box.

|  |
| --- |
| The run result |
| A screenshot of a computer screen  Description automatically generated |

# P3 – Perfect Numbers

An integer number is said to be a perfect number if its factors, including 1 (but not the number itself), sum to the number.

For example, **6** is a perfect number, because **6 = 1 + 2 + 3**.

Write a function **isPerfect** that determines whether parameter number is a perfect number.

Use this method in an program that displays all the perfect numbers between 1 and 1,000.

Display the factors of each perfect number to confirm that the number is indeed perfect.

|  |
| --- |
| Your C++ code for this problem |
| #include <iostream>  #include <string>  using std::cout;  using std::endl;  using std::cin;  bool isPerfect(unsigned n)  {  bool retVal = false;  int sumOfFactors = 0;  for (int i = 1; i <= n / 2; i++)  {  if (n % i == 0)  {  sumOfFactors = sumOfFactors + i;  }  if (sumOfFactors == n)  {  retVal = true;  break;  }  }  return retVal;  }  int main()  {  for (int n = 1; n <= 1000; n++)  {  if (isPerfect(n))  {  cout << n << " is Perfect \n Factor are: ";  for (int i = 1; i <= n / 2; i++)  {  if (n % i == 0)  cout << i;  }  cout << endl;  }  else  {  // save the planet -  // assume silence means not perfect  }  }  return 0;  } |

Run the code and insert the result in the following box.

|  |
| --- |
| The run result |
| A screenshot of a computer screen  Description automatically generated |

# P4 – Assign Grades

Using an array of size 100. write a program that reads student scores, gets the best score, and then assigns grades based on the following scheme:

Grade is A if score is 7 = best - 10;

Grade is B if score is 7 = best - 20;

Grade is C if score is 7 = best - 30;

Grade is D if score is 7 = best - 40;

Grade is F otherwise.

The program prompts the user to enter the total number of students, then prompts the user to enter all of the scores, and concludes by displaying the grades. Here is a sample run:

|  |
| --- |
| Enter the number of students: 4  Enter 4 scores: 40 55 70 58  Student 0 score is 40 and grade is C  Student 1 score is 55 and grade is B  Student 2 score is 70 and grade is A  Student 3 score is 58 and grade is B |

|  |
| --- |
| Your C++ code for this problem |
| #include <iostream>  #include <string>  using std::cout;  using std::endl;  using std::cin;  int main()  {  const int MAX\_NUMBER\_OF\_STUDENTS = 100;  int scores[MAX\_NUMBER\_OF\_STUDENTS];  int best = 0;  int numberOfScores;  cout << "Enter the number of students: ";  cin >> numberOfScores;  cout << "Enter " << numberOfScores << " score: ";  for (int i = 0; i < numberOfScores; i++)  {  cin >> scores[i];  if (scores[i] > best)  best = scores[i];  }  char grade;  for (int i = 0; i < numberOfScores; i++)  {  if (scores[i] >= best - 10)  {  grade = 'A';  }  else if (scores[i] >= best - 20)  grade = 'B';  else if (scores[i] >= best - 30)  grade = 'C';  else if (scores[i] >= best - 40)  grade = 'D';  else  grade = 'F';  cout << "Student# " << i + 1 << "grade is " << grade << endl;  }  return 0;  } |

Run the code and insert the result in the following box.

|  |
| --- |
| The run result |
| A screenshot of a computer screen  Description automatically generated |

# P5 – Pattern Recognition: Consecutive Four Equal Numbers

Write the following function that tests whether the array has four consecutive numbers with the same value.

**bool** isConsecutiveFour(**const** **int** values[], **int** size)

Write a test program that prompts the user to enter a series of integers and displays if the series contains four consecutive numbers with the same value.

Your program should first prompt the user to enter the input size—i.e., the number of values in the series. Assume the maximum number of values is 80.

Here are sample runs:

|  |
| --- |
| Enter the number of values: 8  Enter the values: 3 4 5 5 5 5 4 5  The list has consecutive fours |

|  |
| --- |
| Enter the number of values: 9  Enter the values: 3 4 5 5 6 5 5 4 5  The list has no consecutive fours |

|  |
| --- |
| Your C++ code for this problem |
| #include <iostream>  #include <string>  using std::cout;  using std::endl;  using std::cin;  bool isConsecutiveFour(const int values[], int size)  {  for (size\_t i = 0; i < size - 3; i++)  {  bool isEqual = true;    for (int j = i; j < i + 3; j++)  {  if (values[j] != values[j + 1])  {  isEqual = false;  break;  }  }  if (isEqual)  return true;  }  }  int main()  {  const int MAX\_SIZE = 80;  cout << "Enter the number of values: ";  int size;  cin >> size;  int values[MAX\_SIZE];  cout << "Enter the values: ";  for (int i = 0; i < size; i++)  cin >> values[i];  if (isConsecutiveFour(values, size))  {  cout << "List had consecutive numbers " << endl;  }  else  {  }  return 0;  } |

Run the code and insert the result in the following box.

|  |
| --- |
| The run result |
| A screenshot of a computer screen  Description automatically generated  A screenshot of a computer screen  Description automatically generated |

# P6 – Common Elements

Write a program that prompts the user to enter two arrays of 10 integers and displays the common elements that appear in both arrays. Here is a sample run.

|  |
| --- |
| Enter list1: 8 5 10 1 6 16 61 9 11 2  Enter list2: 4 2 3 10 3 34 35 67 3 1  The common elements are 10 1 2 |

|  |
| --- |
| Your C++ code for this problem |
| #include <iostream>  #include <string>  using std::cout;  using std::endl;  using std::cin;  int main()  {  int list1[10];  int list2[10];    cout << "Enter list1: ";  for (int i = 0; i < 10; i++)  cin >> list1[i];  cout << "Enter list2: ";  for (int i = 0; i < 10; i++)  cin >> list2[i];    for (unsigned outer = 0; outer < 10; outer++)  {  int out = list1[outer];  for (unsigned inner = 0; inner < 10; inner++)  {  if (out == list2[inner])  cout << out << " ";  }  }  return 0;  } |

Run the code and insert the result in the following box.

|  |
| --- |
| The run result |
| A screenshot of a computer screen  Description automatically generated |

**The end**