# 

**Iqra National University Peshawar Pakistan**

**Department of Computer Science**

Spring Semester, Final Term Exam, July 2020

|  |  |  |  |
| --- | --- | --- | --- |
| Paper : | **Programming Fundamentals (Lab)** | Date and Starting Time: | **09/July/2020, 09:00 am** |
| Program: | **BS (CS & SE)** | Uploading Date and End Time: | **09/July/2020, 3:00 pm** |
| Teacher Name: | **Dr. Fazal-e-Malik** | Marks | **100** |

**Name: Haseeb Javed**

**ID: 16890**

**Note: Attempt all Questions.**

|  |  |
| --- | --- |
| Q. 1 | Read A, B and C representing the three sides of a triangle. Write a program to find out its area the formula is given below:  Where S=  **ANS) C++ CODE:**  #include <bits/stdc++.h>  using namespace std;    float findArea(float a, float b, float c)  {  // Length of sides must be positive  // and sum of any two sides  // must be smaller than third side.  if (a < 0 || b < 0 || c < 0 ||  (a + b <= c) || a + c <= b ||  b + c <= a)  {  cout << "Not a valid trianglen";  exit(0);  }  float s = (a + b + c) / 2;  return sqrt(s \* (s - a) \*  (s - b) \* (s - c));  }    // Driver Code  int main()  {  float a = 3.0;  float b = 4.0;  float c = 5.0;    cout << "Area is " << findArea(a, b, c);  return 0;  } |
|  |  |
| Q. 2 | Write a C++ program to get marks obtained by a student in percentage ***P*** and then find the division according to the below rules:   * If Percentage P is above or equal to 60 then display…………..1st Division. * If Percentage Pis between 50 & 59 then display…………………2nd Division. * If Percentage P is between 40 & 49 then display….……………3rd Division. * If Percentage P is less than 40 then display………………………Fail.   **ANS) C++ CODE:**  #include <stdio.h>  int main()  {  int phy, chem, bio, math, comp;  float per;  /\* Input marks of five subjects from user \*/  printf("Enter five subjects marks: ");  scanf("%d%d%d%d%d", &phy, &chem, &bio, &math, &comp);  /\* Calculate percentage \*/  per = (phy + chem + bio + math + comp) / 5.0;  printf("Percentage = %.2f\n", per);  /\* Find division according to the percentage \*/  if(per >= 70)  {  printf("1st Division");  }  else if(per >= 60)  {  printf("2nd Division");  }  else if(per >= 50)  {  printf("3rd Division");  }  else if(per >= 40)  {  printf("Fail");  return 0;  }  } |
|  |  |
| Q. 3 | Write a C++ program to convert 5 feet to the equivalent number of (a) Inches (b) Yards. Where 1foot =12 Inches and 1 yard=3 feet)  **ANS) C++ CODE:**  #include <iostream>  using namespace std;  int  main ()  {  int inches;  int feet;  int yards;  cout << "Number of Inches\n";  cin >> inches;  cout << "Number of Yards is\n";  yards = inches % 36;  cout << yards;  cout << "number of feet\n";  feet = inches % 12;  cout << feet;  cout << "number of inches\n";  cout << inches;  yards = inches / 36;  cout << yards;  return 0;  } |
|  |  |
| Q.4 | Write a C++ program to find the sum of the following series:  **2+4+6+8+10**  **ANS) C++ CODE:**  #include <iostream>  using namespace std;  int main()  {  int i, n, sum = 0;  cout << "\n\n Find the sum of the series 2+4+6+8+10 (n+n):\n";  cout << "------------------------------------------------------------------------------------\n";  cout << " Input the value for nth term: ";  cin >> n;  for (i = 1; i <= n; i++)  {  sum += i + i;  cout << i << "+" << i << " = " << i + i << endl;  }  cout << " The sum of the above series is: " << sum << endl;  } |
|  |  |
| Q.5 | Write a C++ program to input Hours Worked and Hour Rate of an Employee. Calculate and display the Gross-Pay, Tax and Net-Pay; where  Gross-Pay=Hour-Worked\*Hour-Rate  Tax=10% of Gross-Pay  Net-Pay=Gross-Pay – Tax  **ANS) C++ CODE:**  #include <iostream>  #include <iomanip>  using namespace std;  // Declare Functions  double computeGross( double hoursWorked, double hourlyWage);  double computeDeductions(double grossPay);  double computeNet( double grossPay, double deductions);  void validateHours(double hoursWorked);  void validateWage(double hourlyWage);  int main()  {  // Declare Variables  double hoursWorked = 0;  double hourlyWage = 0;  double grossPay = 0;  double deductions = 0;  double netSalary = 0;  // Get the hours worked and hourly wage  cout << "Please enter the amount of hours worked (HH.MM): " << endl;  cin >> hoursWorked;  cout << "Please enter in your hourly wage: $" << endl;  cin >> hourlyWage;  validateHours (hoursWorked);  validateWage(hourlyWage);  grossPay = computeGross(hoursWorked, hourlyWage);  deductions = computeDeductions(grossPay);  netSalary = computeNet(grossPay, deductions );  // Output the results  cout << fixed << setprecision(2)  << "The net salary is: $" << netSalary << endl;  return 0;  }  // compteGross() function - get gross salary based on hours worked and hourly wage.  double computeGross(double hoursWorked, double hourlyWage)  {  return hoursWorked \* hourlyWage;  }  // computeDeductions() function - gets salary and calculates deductions  double computeDeductions(double grossPay)  {  double deductions;  if(grossPay < 2500)  {  deductions = (grossPay \* .10) \* .175;  }  else  {  deductions = (grossPay \* .20) \* .175;  }  return deductions;  }  // computeNet() function - prints out gross salary,total deductions and net salary  double computeNet(double grossPay, double deductions)  {  double netSalary;  netSalary = grossPay - deductions;  cout << "The gross salary is: $" << grossPay << endl;  cout << "The total deductions are: $" << deductions << endl;  cout << "The net salary is: $" << netSalary << endl;  return netSalary;  }  // validateHours() function - input validation; hours worked can;t exceed 150 or be neg.  void validateHours(double hoursWorked)  {  if(hoursWorked < 0 || hoursWorked > 150)  {  cout << "Error! Hours can't be negative or exceed 150\n";  }  }  // validateWage() - Input validation; wage can't exceed 200 or be negative  void validateWage(double hourlyWage)  {  if(hourlyWage < 0 || hourlyWage > 200)  {  cout << "Error! Wage can't be negative or exceed 200\n";  }  } |
|  |  |

☺ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*☺

***Note: Please write your Name and ID on top of your answer Paper otherwise you will get zero marks.***