|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **National University of Computer and Emerging Sciences, Lahore Campus** | | | | |
| C:\Users\saif\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.Word\final design.jpg | **Course Name:** | **Programming Fundamentals** | **Course Code:** | **CS** |
| **Program:** | **CS** | **Semester:** | **Fall 2018** |
| **Duration:** | **60 Minutes** | **Total Marks:** | **20** |
| **Paper Date:** |  | **Weight** | **15** |
| **Section:** | **ALL** | **Page(s):** | **2** |
| **Exam Type:** | **Sessional - II** |  |  |
| **Student : Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Roll No.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Section:\_\_\_\_\_\_\_** | | | | |
| **Instruction/Notes:** | Solve the exam on this question paper. You may use rough sheets, but they must not be attached. | | | |

**Problem 1** **[ 5 pts]** An **append** function is given below which takestwo strings as parameters and it should append data of string s2 after s1**.** For example, if s1 = “Ali”, and s2 = “Ahmad”, then after function call s1 should be updated as “AliAhmad”. Identify the logical errors in the code below **by circling them**, and provide corrections based on the above description.

|  |  |
| --- | --- |
| #include<iostream> | Specify corrections for respective lines |
| using namespace std; | here! |
| void append(char s1 [], char s2 []){ |  |
| int l1 = 0, l2 = 0, |  |
| for (; s1[l1] != '\0'; l1++); |  |
| for (; s2[l2] != '\0'; l2++); |  |
| for (int i = l1+1, j=0; j<l2; i++) | for (int i = l1, j=0; j<l2; i++) |
| s1[i++] = s2[j++]; | s1[**i**] = s2[j++]; |
| s1[l1 + l2 + 1] = '\0'; | s1[l1 + l2 ] = '\0'; |
| } |  |

**Problem 2 [5 pts]** **What is the output of the following code?**

|  |  |
| --- | --- |
| void doSomething(int d[][4], int dim, int num)  {  int size = dim;  for (int i = 0; i < dim; i++) {  int m = 0;  for (int j=1; j<size; j++)  {  if (d[num][m] < d[num][j])  m = j;  }  swap(d[num][m], d[num][size - 1]);  size--;  }  } | int main()  {  int nums[4][4] = {{26,19,14,15},  {2,3,4,1},{17,30,18,1},{1,2,6,5} };  doSomething(nums, 4, 0);  doSomething(nums, 4, 3);  for (int i = 0; i < 4; i++)  {  for (int j = 0; j < 4; j++)  cout << nums[i][j]<<" ";  cout << endl;  }  } |

Output:

14 15 19 26

2 3 4 1

17 30 18 1

1 2 5 6

**Problem 3** **[10 pts]** Write a function called **maxDifferenceWithinK**. It takes an integer array A, its size n, and a number k, as parameters. It finds a pair of numbers x and y in A (where x comes before y in A), such that that they have the maximum difference given by **y-x.** Additionally, the difference between the indices of x and y should be <= **k**. The function reports three things: the index of x, the index of y, and the difference **y-x**. As an example, consider A as

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 77 | 88 | 11 | 3 | 1 | 5 | 6 | 30 | 23 | 0 |

**Sample example1**: input n = 10 and k=2

**Output**: 5, 7, 25. How? x and y at indexes 5 and 7 respectively, have values 5 and 30 with difference of 25. Difference between indices is 7-5 = 2 which is <= k.

**Sample example2**: input n = 10 and k=5

**Output**: 4, 7, 29. How? x and y at indexes 4 and 7 respectively, have values 1 and 30 with difference of 29. Difference between indices is 7-4 = 3 which is <= 5 or k.

|  |
| --- |
| int maxDiff = data[1] - data[0]; //important to initialize maxDiff.  int st = 0;  int end = 1;  for (int i = 0; i < size-k; i++) //start at index i  {  for (int j = i + 1; j < i + k; j++) //and calculate its difference with the next k elements  {  if(maxDiff < data[j]-data[i]) //if the difference is > than maxDiff update maxDiff  {  maxDiff = data[j] - data[i];  st = i; //store the start and end indices.  end = j;  }  }  }  cout << st << ", " << end << ", " << maxDiff;  } |