Grade 11 Questions

Good day Grade 11 Learners

You may complete these two questions below by today evening and send to me via email. You may design the form using suitable buttons and write the code inside the different buttons instead of writing Procedure or Function.

**QUESTION ONE**

A T-shirt manufacturer is designing a new T-shirt in just one size. He conducts a

sample survey of the market to reveal what would be the best T-shirt size to manufacture.

(*T-shirt sizes could be any number in the range: 3 to 10)*

**Using the following declaration**

TYPE List = array[1..100] of integer;

VAR Sizes:List;

**You are required to write a complete Pascal program that uses modular**

**programming and parameter passing throughout to do the following:**

* 1. Write a **procedure** named **Enter** to read in the survey data into a one

dimensional array. Input is terminated by -1. (5)

* 1. Write a **function** named **Cal** to determine the number of persons with a

T-shirt size above the average T-shirt size. (8)

* 1. Write a **procedure** named **Best** to determine the best size to manufacture.

The best size is the size with the highest occurrence in the data list. Assume that

there is only one size with the highest occurrence in the survey. (9)

* 1. Write a **procedure** named **Display** to print the list of sizes, the number of T-shirt

sizes above the average size and the best size to manufacture. Function **Cal** and

Procedure **Best** are called from Procedure **Display**. (4)

* 1. Ensure that the main program is written. (1)

**SAMPLE INPUT**

Enter T-shirt sizes:

7 5 4 7 4 5 4 8 7 4 5 7 5 9 5 8 7 4 7 4 7 9 -1

**SAMPLE OUTPUT**

List of T-shirt sizes:

7 5 4 7 4 5 4 8 7 4 5 7 5 9 5 8 7 4 7 4 7 9

Number of sizes above the average size : 11

Best size to manufacture : 7

STYLE : (3)

**[30]**

**QUESTION TWO**

***N.B. Write the complete Pascal program. Use local variables****.*

Declare a global one-dimensional array called **Num** to hold 20 integer values.

* 1. Write a **procedure** called **FormArray**
* to read in N (where N is the size of the array). No data validation is

required.

* generate N random numbers in the range 1 to 100 and store

them in the array. (5)

* 1. Write a **procedure** called **Display** to output the array elements next to each

other using suitable spacing between each element. (2)

2.3 Write a **procedure** called **Sort** to sort the elements in ascending order. (6)

2.4 Write a **function** called **Prime** to return the Boolean value true if the size of

the array (N) is a prime number.

**(*N.B. A prime number is a number that has two factors only*)** (4)

* 1. Write a **procedure** called **Split** that uses two locally declared arrays to

separate the odd numbers from the even numbers and rearrange the original

array so that the even numbers appear first followed by the odd numbers. (7)

* 1. Write the **main body** of the program calling the appropriate procedures to

display the original array, the sorted array and the processed array below each

other. Also display a message indicating whether the size of the array (N) is

a prime number or not. (6)

**Sample Input**

Enter size of array:

15

**Sample Output**

Original: 34 89 34 17 53 41 52 55 88 37 18 30 45 96 48

Sorted: 17 18 30 34 34 37 41 45 48 52 53 55 88 89 96

Processed: 18 30 34 34 48 52 88 96 17 37 41 45 53 55 89

15 is not a prime number  **[30]**