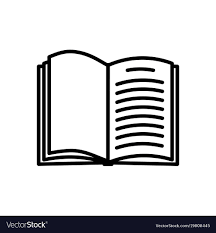
**Performance Tasks**

|  |
| --- |
| ***Performance Task 1***  **Directions: Answer the ff questions**   1. It is an Array property which is used to get the total number of elements in all the dimensions of the Array. **Length Property** 2. The first element of an array can be found in index \_\_0\_\_. 3. *Array indices are of type \_\_integers\_\_\_.* 4. An Array is a collection of \_\_\_\_multiple\_\_\_\_ elements (the same type). 5. Component of the by using its \_index\_\_\_\_. ***Index***   ***Perform the ff using the given Array below.***  **Given : int**[] arr = **new** **int**[6] { 5, 8, 9, 25, 0, 7 };   1. (arr[4] + arr[0 )] \* arr[2] 45 2. arr[5] - (arr[3] /arr[0]) 2 3. Address of the first element 8 4. Address of the last element 7 5. Length of the arr[] 6 |
| ***Performance Task 2***    **Direction:**      using System;      class Program      {          public static void Main(string[] args)          {              int[] arr1 = new int[5];              int[] arr2 = new int[5];              int[] arr3 = new int[5];          Console.WriteLine("==ARRAY PROGRAM==");          for (int i=0;i<5;i++)          {              Console.Write("Enter Element {0}: ", i );              arr1[i] = Convert.ToInt32(Console.ReadLine());           }            int j=0, k=0;              for (int i=0;i<arr1.Length;i++)              {                  if ((arr1[i]%2)==0)                  {                      arr2[j] = arr1 [i];                      j++;                    }                  else                  {                      arr3[k] = arr1[i];                      k++;                  }              }              Console.Write("Odd Numbers ");              for (int i=0;i<k;i++)              {                  Console.Write("{0}, ", arr3[i]);              }              Console.Write("\nEven Numbers ");              for (int i =0; i<j;i++)              {                  Console.Write("{0}, ", arr2[i] );              }           }      }    *Screenshot of Sample Output* |
|  |

**Understanding Directed Assessment**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | ***Criteria*** | ***1*** | ***2*** | ***3*** | ***4*** | | ***Delivery*** | * Completed less than 75% of the requirements * Not delivered on time or not in correct format (disk, email, Canvas, printout, etc.) * Does not comply with requirements (does something other than requirements) | * Completed at least 75% of the requirements * Delivered on time, and in correct format | * Completed between 80-99% of the requirements | * Completed 100% of requirements | | ***Coding Standards*** | * No programmer name included * Poor use of white space (indentation, blank lines) making code hard to read * Disorganized and messy * Uses global variable(s), goto/continue/ exit/ break (except in switch) * Ambiguous identifiers | * Includes name, and assignment title * White space makes program fairly easy to read * Organized work * Good use of variables | * Good use of white space * Organized work * Good use of variables and constants * Minimum line-wrap | * Excellent use of white space * Creatively organized work * Excellent use of variables and constants * No magic numbers * Correct identifiers for constants * No line-wrap | | ***Runtime*** | * Does not execute due to syntax errors * Does not execute due to runtime errors (endless loop, crashes, etc.) * User prompts are misleading or non-existent * No testing has been completed | * Executes without errors * User prompts contain little information, poor design * Some testing has been completed | * Executes without errors * User prompts are understandable, minimum use of symbols or spacing in output * Thorough testing has been completed | * Executes without errors excellent user prompts, good use of symbols, spacing in output * Thorough and organized testing has been completed and output from test cases is included | | ***Efficiency*** | * A difficult and inefficient solution | * A logical solution that is easy to follow but it is not the most efficient | * Solution is efficient and easy to follow (i.e. no confusing tricks) | * Solution is efficient, easy to understand, and maintain | |

**Learning Resources** 

|  |
| --- |
| * Students’ Handbook * Karumanchi, N. (2017). *Data Structures and Algorithms Made Easy* * Wengrow, J. (2017). *A Common-Sense Guide to Data Structures and Algorithms* * https://www.tutorialsteacher.com/csharp/csharp-list * https://www.dotnetperls.com/list * https://www.geeksforgeeks.org/hashset-in-c-sharp-with-examples/ |

