CS10003 : Programming and Data Structures (Spring 2023-2024)

Class Test 2

21-Mar-2024 (Thursday) Maximum Marks: 20 06:45pm – 07:45pm

Roll: _____ **Name:** _____

[*Write your answers in question paper. Answer all questions. All questions use C programming language.*]

Q1. Fill in the blanks for the following C function which returns the total number of duplicate elements (i.e., the count of elements that appears more than once) in an array of integers. **(4)**

```
_____int_____ countDuplicate ( int _____arr[]_____ , int num ) {
  int i, j, count = 0;
  for ( i = 0 ; _____i < num_____ ; _____++i_____ ) {
      for ( j = _____i+1_____ ; _____j < num_____ ; ++j ) {
          if ( _____arr[i] == arr[j]_____ )   { count++; break; }
      }
  }
  return _____count_____;
}
```

Q2. Fill in the blanks for the following C function `changeToPalin` so that it can return the minimum number of character replacements required to turn a string into a palindrome as well as modify the provided string into the desired palindrome internally. Note that, a palindrome is a string that reads the same both forwards and backwards. Examples of palindromes are "noon" and "deified". Hence,

- The minimum replacements required for "noon" and "deified" are *zero* character (each).
- The string "civil" can be turned into a palindrome by replacing minimum *one* character.
- The string "rastor" can be turned into a palindrome by replacing minimum *two* characters.
- The string "drawing" can be turned into a palindrome by replacing minimum *three* characters. **(3)**

```
#include <stdio.h>
#include <string.h>
int changeToPalin ( char *str, int len ) {
  if ( _____len <= 1_____ )   return 0;   /* base condition */
  if ( str[0] == str[len-1] )
      return _____changeToPalin ( str+1, len-2 )_____ ;
  str[0] = str[len-1];   /* modifying the string into palindrome */
  return _____1 + changeToPalin ( str+1, len-2 )_____ ;
}
int main ( ) {
  char S[100];
  printf ("Enter String:  ");   scanf ("%s", S);
  printf ( "%d ", changeToPalin(S, strlen(S)) );   printf ("%s", S);
  return 0;
}
```

In case that your `changeToPalin` function can produce the desired result as specified in the problem statement, what will be the output of the above C program when the input string is "`neuralnet`"? **(2)**

3 tenlalnet

Q3. You are given a set of nuts and bolts, each with unique sizes. However, the sizes are not directly comparable with each other; you can only attempt to match a nut with a bolt to determine if one is larger, smaller, or equal to the other. The following C program takes arrays of nuts and bolts as input, matches each nut with its corresponding bolt, and prints out the matched pairs. The desired result of the program is to correctly pair each nut with its corresponding bolt. Some sections of the code are left blank. Fill in these blanks to ensure the code produces the desired results. **(5)**

```
#include <stdio.h>
#define   MAX   6
void printArray ( char arr[] ) {
  for(int i = 0; i < MAX; i++)   printf ("%c ", arr[i]);
}
void swap ( char arr[], int i, int j ) {
  int temp = arr[i];   arr[i] = arr[j];   arr[j] = temp;
}
int splitArray ( char arr[], int low, int high, char pivot ) {
  int i = low, j;
  for( j = _____low_____ ; _____j < high_____ ; j++ ) {
      if ( _____arr[j] < pivot_____ ) {
          swap ( arr, i, j );   i = _____i + 1_____ ;
      }
      else if ( _____arr[j] == pivot_____ ) {
          swap ( arr, j, high );   j = _____j - 1_____ ;
      }
  }
  swap ( _____arr, i, high_____ ) ;
  return i;
}
void matchPairs ( char nuts[], char bolts[], int low, int high ) {
  int pivot, tovip;
  if ( _____low < high_____ ) {
      pivot = splitArray ( nuts, low, high, bolts[high] );
      tovip = splitArray ( bolts, low, high, nuts[pivot] );
      matchPairs ( _____nuts, bolts, low, pivot-1_____ );
      matchPairs ( _____nuts, bolts, pivot+1, high_____ );
  }
}
int main ( ) {
  char nuts[MAX] = { '@', '#', '$', '%', '^', '&' };
  char bolts[MAX] = { '$', '%', '&', '^', '@', '#' };
  matchPairs ( nuts, bolts, 0, MAX-1 );
  printf ("Matched nuts and bolts are:\n");
  printArray(nuts);   printf("\n");   printArray(bolts);
  return 0;
}
```

Q4. If the following C programs terminate, then write the output (when no output can be generated, write `NO-OUTPUT`). Otherwise, say that the program does not terminate (write `NO-TERMINATION`).

Assume that, a `pointer` *and an* `integer` *variables take* 8 *and* 4 *bytes of storage space, respectively.* **(2×3)**

```
#include <stdio.h>

int main ()
{
  int a = 1, b = 1, c = 1;
  if (a == b == c)  printf ("U ");
  if (a != b != c)  printf ("V ");
  if ( a < b < c )  printf ("W ");
  if ( a > b > c )  printf ("X ");
  if ( a < b > c )  printf ("Y ");
  if ( a > b < c )  printf ("Z ");
  return 0;
}
```

Answer: U V W Z

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
  int *A[3], i;
  int B[] = {1, 2, 3};
  int C[][3] = { {1, 2, 3},
                 {4, 5, 6},
                 {7, 8, 9} };
  for ( i=0; i<3; i++ )
    A[i] = (int *) malloc (10 *
                  sizeof(int));
  printf ("%u ", sizeof(A[1]));
  printf ("%u ", sizeof(A));
  printf ("%u ", sizeof(B));
  printf ("%u ", sizeof(C));
  return 0;
}
```

Answer: 8 24 12 36

```
#include <stdio.h>

void genPat (int A[][6], int n)
{
  int i, j, num = n;
  for ( i=0; i<n+1; i++ ) {
    for ( j=0; j<num; j++ )
      A[i][j] = 1;
    for ( j=num; j<num + n; j++ )
      A[i][j] = 2;
    for ( j=num+n; j<2*n; j++ )
      A[i][j] = 1;
    --num;
  }
}

void printArr (int A[][6], int n)
{
  int i,j;
  for ( i=0; i<n+1; i++ ) {
    for ( j=0; j<2*n; j++ )
      printf ("%d", A[i][j]);
    printf("\n");
  }
}

int main ()
{
  int A[6][6];
  genPat(A,3);
  printArr(A,3);
  return 0;
}
```

Answer:
111222
112221
122211
222111

— The question paper ends here. —