```
+=================================================================+
|           Indian Institute of Technology Kharagpur              |
|        Department of Computer Science and Engineering           |
|-----------------------------------------------------------------|
|       CS10003 : Programming and Data Structures (Theory)        |
|    Autumn 2020   |   Long Test 2  (ALL Q&A)   |   Marks: 40     |
| Date: 10-Feb-2021 (Wednesday) | Time: 09:00am-10:15am (75 min) |
+=================================================================+

   Instruction: All LT2 Questions and Answers can be found here.


================================================================================
++++++++++++++++++++++ Topic-1: Functions and Recursions ++++++++++++++++++++++


******** Question ********
Fill in the following blanks to print whether an input matrix is a symmetric mat
rix or not. A matrix is called as symmetric if its (i,j) element is same as (j,i
) for all i,j.

void symmMat( ____(1)____, int row, int col)
{
   int i, j;
   if( ____(2)____ ) printf("Not Symmetric");
   for(i=0;i<row;i++) {
      for(j=i+1;j<__(3)__;j++) {
         if(mat[i][j] != mat[j][i]) {
            ____(4)____;
            break;
         }
      }
   }
   ( ____(5)____ )? printf("Symmetric"):printf("Not Symmetric");
}

******** Answer ********
(1) int mat[100][100]
// Note: 100 can be any const integer; it can also be int mat[][100]
(2) row!=col
(3) col
(4) i=row
(5) i==row


******** Question ********
Fill in the following blanks to complete the following function.

fn(m,n) = m, if m==n
fn(m,n) = m, if n==1
fn(m,n) = 1, if n==0
fn(m,n) = fn(m-1,n) + fn(m-1,n-1), otherwise

____(1)____ fn(int m, int n)
{
    if( ____(2)____ ) return( ____(3)____ );
    if( ____(4)____ ) return (1);
    else return ( ____(5)____ );
}
```

******** Answer ********
(1) int
(2) m==n || n==1
(3) m
(4) n==0
(5) fn(m-1,n) + fn(m-1,n-1)

******** Question ********
Two integer numbers are read as two arrays of characters. Fill in the following blanks to return 1, or 0, or -1, whenever x is greater than, or equal to, or les s than y, respectively.

```
int compare(char x[],char y[])
{
    int num1,num2,i;
    ____(1)____ ;
    do {
        num1 = ____(2)____ + ____(3)____ ;
        i++;
    } while(x[i]!='\0') ;
    i=0;
    do {
        num2 = ____(4)____ + ____(5)____ ;
        i++;
    } while(y[i]!='\0') ;

    if(num1>num2) return 1;
    else if(num1<num2) return -1;
    else return 0;
}
```

******** Answer ********
(1) num1=num2=i=0
(2) num1*10
(3) (x[i]-'0')
(4) num1*10
(5) (y[i]-'0')

******** Question ********
Fill in the following blanks to complete a recursive function power() that takes in a positive integer n, a non-negative integer p and returns n raised to the p ower p.
The call power(2, 5) should return 32.

```
____(1)____ power( ____(2)____ )
{
  if( ____(3)____ )
    return ____(4)____ ;
  else
    return ____(5)____ ;
}
```

******** Answer ********
(1) int
(2) int n, int p (the first argument is the base, and the second argument is the exponent)
(3) p==0
(4) 1

(5) n*power(n, p-1)

In the answers, names of the variables may be different.

******** Question ********
Fill in the following blanks to complete the following function eseries that on input n approximates e^x by the polynomial 1 + x + x^2/2! + x^3/3! + ... + x^9/9!
(a^b reads as, a to-the-power b)

```
double eseries(double n)
{
  double i, sum=1, term=1;
  for(i=____(1)____ ; i<= ____(2)____ ; ____(3)____ )
  {
    term = ____(4)____ ;
    sum += ____(5)____ ;
  }
  return sum;
}
```

******** Answer ********
(1) 1
(2) 9
(3) i++
(4) term*n/i
(5) term

******** Question ********
Consider the following function.

```
void f(int m)
{
  static int n=2;
  printf("%d, ", n=m*n);
}
```

What is printed if f is called from the main 5 times with inputs 2, 5, 3, 1 and -1 respectively?

******** Answer ********
4, 20, 60, 60, -60,

Marking scheme: 1 mark for each number printed. Minor issues involving the terminal comma, spaces, etc can be neglected.

******** Question ********
You are given a binary pattern containing wildcard character '?' at few positions. The objective is to find all possible combinations of binary strings that can be formed by replacing the wildcard character by either '0' or '1'. For example, for wildcard pattern '10?10?1?', the possible combinations are
10010010
10010011
10010110
10010111
10110010
10110011
10110110
10110111

You need to complete the recursive solution provided below in order to achieve t
he objective. The objective of the recursive solution is -- first to replace ‘?’
 by 0 and proceed to the next character, second do the same set of steps by repl
acing ‘?’ with 1.

```
#include <stdio.h>

void printAllCombinations(char pattern[], int i) {
    int k;
    if (pattern[i] == ____(1)____) {
        printf("%s\n", pattern);
        return;
    }
    if (pattern[i] == '?') {
        for (k = 0; ____(2)____; k++) {
            pattern[i] = k + '0';
            printAllCombinations(____(3)____);
            pattern[i] = ____(4)____;
        }
        return;
    }
    printAllCombinations(____(5)____);
}

int main() {
    char pattern[] = "10?10?1?";
    printAllCombinations(pattern, 0);
    return 0;
}
```

******** Answer ********
(1) '\0'
(2) k < 2
(3) pattern, i + 1
(4) '?'
(5) pattern, i + 1

******** Question ********
For an input array of integers, print the output in the specified format where t
he lowest level prints all the array elements. In a bottom-up approach, at each
 level, the total number of elements is one less than the previous level (given i
n next line as we are looking bottom-up) and elements at that level is the sum o
f the consecutive two elements in the previous level (given in next line as we a
re looking bottom-up). You need to complete the blanks in the code provided belo
w in order to construct the recursive solution to achieve the above-mentioned ob
jective.

Let us give a demonstration of the process in the following.

Input:
Enter array size: 5
Enter input: 4
Enter input: 2
Enter input: 1
Enter input: 7
Enter input: 6

Output:

```
52
20,  32
9,  11,  21
6,  3,  8,  13
4,  2,  1,  7,  6

Explanation :
[52]                 -->        (20 + 32 = 52)
[20,  32]            -->        (9 + 11 = 20, 11 + 21 = 32
[9,  11,  21]        -->        (6 + 3 = 9, 3 + 8 = 11, 8 + 13 = 21)
[6,  3,  8,  13]     -->        (4 + 2 = 6, 2 + 1 = 3, 1 + 7 = 8, 7 + 6 = 13)
[4,  2,  1,  7,  6]

void printOutput(int A[] , int n)
{
    // Base case
    if( ____(1)____ )
    {
        ____(2)____ ;
    }

    int *temp;
    temp=(int *)malloc((n-1)*sizeof(int));

    for (int i = 0; i < n - 1; i++)
    {
        int x = ____(3)____ ;
        temp[i] = x;
    }

    // recursive call
    ____(4)____ ;

    for (int i = 0; i < n ; i++)
    {
        if( ____(5)____ )
            printf("%d ",A[i]);
        else
            printf("%d, ",A[i]);
    }
    printf("\n");
}
void main()
{
    int n;
    printf("Enter array size: ");
    scanf("%d",&n);

    int A[n];

    for(int i = 0; i < n ;i++)
    {
       printf("Enter input: ");
       scanf("%d",&A[i]);
    }

    printf("\n");
    printOutput(A, n);
}
```

******** Answer ********
(1) n < 1
(2) return
(3) A[i] + A[i + 1]
(4) printOutput(temp, n - 1)
(5) i == n - 1

******** Question ********
You are given the following pattern

```
* * * * * * *
 * * * * * *
  * * * * *
   * * * *
    * * *
     * *
      *

      *
     * *
    * * *
   * * * *
  * * * * *
 * * * * * *
* * * * * * *
```

Complete the recursive code to print the above pattern.

```
#include<stdio.h>

//recursive function to print a series of stars in a row
void print_star(int n) {
    if(n == 0)
        return;
    printf("* ");
    ____(1)____ ;
}

//recursive function to print spaces in a row
void print_space(int n) {
    if(n==0)
        return;
    printf(" ");
    print_space( ____(2)____ );
}

//recursive function to print a row in the pattern. Each row consists of some sp
aces followed by a series of *
void pattern_row(int n, int row) {
    if(row==0)
        return;
    ____(3)____ ;
    print_star(row);
}

//recursive function to print the complete pattern
void pattern(int n, int m) {
    if(m==0)
        return;
    pattern_row(n,m);        //print a single row of the pattern
```

```
    printf("\n");
    pattern( ____(4)____ );  //to print the top triangle of pattern
    printf("\n");            //to print a new line
    ____(5)____ ;            //to print the bottom triangle of pattern
}

int main() {
    int n;
    scanf("%d", &n); //number of rows in one triangle
    pattern(n,n);
    return 0;
}
```

******** Answer ********
(1) print_star(n-1)
(2) n-1
(3) print_space(n-row)
(4) n,m-1
(5) pattern_row(n,m)

******** Question ********
You are given a void function which takes 4 arguments, A, B, C and n. A and B are pointers to two arrays which store two square matrices of size n x n in column major format. Column major format is when a 2D matrix is stored in a 1D array arranged column wise, that is all elements of column 1 are stored first, then column 2 is stored and so on. Your task is to fill in the blanks present in the body of the function so that it stores the multiplication of A and B in array C in a row major format. Assume that there is sufficient contiguous space allocated to C for holding all the elements and all the values in it have been initialized to 0. Do not change the return type or the function signature.

```
void multiply(double *A, double *B, double *C, int n)
{
  for(int i=0; ____(1)____ ; i++) {
    for(int j=0; j < ____(2)____ ; j++) {
      for(int k=0; k<n; k++) {
        C[ ____(3)____ ] += A[ ____(4)____ ] * B[ ____(5)____ ];
      }
    }
  }
}
```

******** Answer ********
(1) i<n
(2) n
(3) i*n+j
(4) i+k*n
(5) k+j*n

******** Question ********
Consider a game played between two players I and II. Initially there is a collection of N coins in a bag from where the two players draw some coins alternately. During a move of Player I, at most M1 number of coins should be taken out from the bag, whereas for a move of Player II, at most M2 number of coins need be taken out. The player who makes the last move and empties the bag wins. The inductive winning rule of the game for player I at some stage of the game when exactly C (C<=N) coins are left in the bag is given below -
(i) If C = 0, Player I cannot make a move and loses.

(ii) If 1 <= C <= M1, Player I can remove all the remaining coins and wins.
(iii) If C > M1, a winning move for Player I is taking out m coins for 1 <= m <= M1 such that (C − m) is not a winning stage for Player II. If such an m exists, Player I will win.

Complete the following recursive function that accepts the current stage C (number of coins left) and M1, M2 as the sole arguments and returns a winning move for player 1 (who is going to make a move), provided that such a move exists. If no winning move exists, the function returns 0.

```
unsigned int winningMove ( unsigned int C , unsigned int M1, unsigned int M2 )
{
  unsigned int m;
  if ( ____(1)____ )
    return 0;
  if ( ____(2)____ )
    return C;

  for (m = 1; m <= M1 ; m++) {
  /* If taking out m coins leaves the opponent in a non-winning stage, */
  if ( winningMove( ____(3)____ , ____(4)____ , ____(5)____ ) == 0 )
    return m;
}
return 0;
}
```

******** Answer ********
(1) C == 0
(2) C <= M1
(3) C-m
(4) M2
(5) M1

******** Question ********
Consider a recursive program that takes the following inputs:
(i) N: a positive integer
(ii) d1: a single digit between 0 and 9
(iii) d2: a single digit (d1!=d2)

The program will recursively print all the numbers that consist of d1 and d2 that are less than equal to N. For example-

Input: N = 15, d1= 1, d2= 5
Output: 15, 11, 5, 1
Input: N = 50, d1= 0, d2= 1
Output: 11, 10, 1, 0
Input: N = 100, d1= 1, d2= 2
Output: 22, 21, 12, 11, 2, 1

Complete the following recursive function.

```
void print_numbers(int N, int d1, int d2)
{
  int status_flag = 1;
  int temp = N;
  if (N > 0) {
      while (temp > 0 && status_flag ==1 ) {
          int digit = ____(1)____ ;
          if (digit != ____(2)____ && digit != ____(3)____ )
              status_flag = 0;
```

```
          temp = ____(4)____ ;
      }
      if ( status_flag == ____(5)____ )
          printf("%d ", N);
      print_numbers(N - 1,d1,d2);
  }
}

int main()
{
    int N, d1, d2;
    scanf(“%d %d %d”, &N, &d1, &d2);
    printNumbers(N,d1,d2);
    return 0;
}
```

******** Answer ********
(1) temp%10
(2) d1
(3) d2
(4) temp/10
(5) 1

==========

==========
++++++++++ Topic-2: Number System and Floating Point Representation ++++++++++

******** Question ********
Convert the following decimal integers to 2’s complement representation (in 16 bits):

(a) 2’s complement of 181 = ______(1)_____
(b) 2’s complement of -181 = ______(2)_____
(c) 2’s complement of -0 = ______(3)_____

Covert following 8 bits 2’s complement number to decimal number:

(d) 2’complement of 11001100 = ______(4)_____ decimal number
(e) 2’complement of 01101001 = ______(5)_____ decimal number

******** Answer ********
(1) 0000000010110101
(2) 1111111101001011
(3) 0000000000000000
(4) -52
(5) 105

******** Question ********
(a) What is the range of signed integer represented by a 16 bit binary number (minimum and maximum)? Assume 2's complement representation is used.
Min: ____(1)____
Max: ____(2)____

(b) How many bits are required if I represent 1234567890 decimal number in 2’s complement representation. Round off to the nearest number of bits that is divisible by 8.

Bits: ____(3)____

(c) What is the range of signed integer represented by the above question [Question (b)] calculated bits (minimum and maximum)? Assume 2’s complement representation is used.
Min: ____(4)____
Max: ____(5)____

******** Answer ********
(1) -32,768
(2) 32,767
(3) 32
(4) -2147483648
(5) 2147483647

******** Question ********
Assume a 16-bit floating point representation.

(a) Convert -24.5 to floating point number. Simple conversion to binary number with integral and fraction part separated by decimal point between ____(1)____ and ____(2)____

24.5 =____(1)____ . ____(2)____

(b) Again assume in a 16 bit representation, MSB is reserved for sign with usual meaning [____(3)____], next 10 bits are reserved for mantissa [____(4)____] and remaining bits are reserved for exponent [____(5)____]. Normalize the mantissa but do not bias the exponent.

24.5 = ____(3)____ ____(4)____ ____(5)____

******** Answer ********
(1) 11000
(2) 1
(3) 0
(4) 1000100000
(5) 00100

******** Question ********
char c1=201, c2=23, c3=-31, c4='L'-'B', c5=0;
Write the contents of the bytes storing each of the 5 characters above in binary, after the above declaration and initialization? Assume 2's complement representation.

******** Answer ********
c1: 11001001
c2: 00010111
c3: 11100001
c4: 00001001
c5: 00000000

Marking scheme: full marks may be awarded if the leading 0's are skipped. For example, the fifth answer can be 0, and the fourth can be 1001.

******** Question ********
What is the representation of -0.15625 in the 32-bit normalized single precision IEEE floating-point format? Write your answer in bits as follows:

sign = ____(1)____
mantissa = ____(2)____
exponent = ____(3)____

******** Answer ********
(1) 1
(2) 01000000000000000000000
(3) 01111100

Marking scheme: 1 mark for the sign bit (1), 2 marks for the exponent (01111100), 2 marks for the mantissa (01000000000000000000000). The grader can go by the visual impression of whether or not the exact number of trailing zero's in the mantissa is correct.

******** Question ********
Answer the following questions:
(1) What is the smallest integer that can be represented by n bit signed 2's complement representation?
(2) What is the largest integer that can be represented by n bit signed 2's complement representation?
(3) What is the smallest integer that can be represented by n bit sign-magnitude representation?
(4) What is the largest integer that can be represented by n bit sign-magnitude representation?
(5) What is the total number of integers that can be represented by n bit signed 1's complement representation?

******** Answer ********
(1) -2^(n-1)
(2) 2^(n-1)-1
(3) -(2^(n-1) - 1)
(4) 2^(n-1) - 1
(5) 2^n - 1

******** Question ********
(A) Consider the 32-bit floating point representation(IEEE 754 format) taught in the class and answer the following questions:

(i) The 32-bit pattern that represents the decimal number (2021) : ____(1)____

(ii) The 32-bit pattern that represents the decimal number (-3.125) : ____(2)____

(iii) The hexadecimal value 0x00000000 corresponds to: ____(3)____

(B) Answer the following questions:
(i) A is a 16-bit signed integer. The 2's complement hexadecimal representation of A is (F87B). The 2's complement representation of 8*A in hexa-decimal format is: ____(4)____

(ii) Let M be the number of distinct 16-bit integers in 2’s complement representation. Let N be the number of distinct 16-bit integers in sign magnitude representation. Then, M–N is = ____(5)____

******** Answer ********
(1) 0 10001001 11111001010000000000000

(2) 1 10000000 10010000000000000000000
(3) the special value +0
(4) C3D8 (in hex)
(5) 1

******** Question ********
Let us consider two numbers, A1 and A2 in IEEE-754 single precision floating point format. A1 = 0x42200000 and A2 = 0xC1200000. The value of exponent E of A2 in 8 bit format is ____(1)____ . So, for A3 = A1/A2, the sign bit of A3 = ____(2)____ , the exponent of value E of A3 in 8 bit format is ____(3)____ , the value of the mantissa of A3 in bit representation is ____(4)____ , the value of A3 in IEEE-754 single precision floating point format is ____(5)____ .

******** Answer ********
(1) 10000010
(2) 1
(3) 10000001
(4) 00000000000000000000000
(5) 0xC0800000

******** Question ********
Let us carry out the addition of two numbers A and B where A = 42.6875 and B = 0.375 in IEEE single precision arithmetic. As both the numbers are positive, the sign bits are 0. Exponent of A and B in binary representation are ____(1)____ and ____(2)____ , while the mantissa are 0101 0101 1000 0000 0000 000 and 1000 0000 0000 0000 0000 000. To add these numbers, the exponents must be the same. Therefore, we make the smaller exponent equal to the larger exponent by shifting the mantissa accordingly. The exponents differ by ____(3)____ (in decimal). Therefore, we must shift the binary point of the smaller number ____(3)____ places to the left.
After shifting the smaller number appropriately, the sum becomes 1 . ____(4)____ (in IEEE single precision format) and the larger number is 1 . 0101 0101 1000 0000 0000 000. You can now perform the binary addition of the two numbers. The sum in decimal format is ____(5)____ .

******** Answer ********
(1) 1000 0100
(2) 0111 1101
(3) 7 or +7
(4) 0101 1000 1000 0000 0000 000
(5) 43.0625

******** Question ********
Consider a C program that inputs two positive integers and compares the number of 1’s byte wise in each number. If the number of 1’s in each byte of the number is equal to that of the second number, the program prints “Matches” otherwise prints “Does not match”.
For example,

Input: 1 32
Output: Matches

Input: 32 1024
Output: Does not match

Complete the following code.

```
int compare_numbers(int num1, int num2)
{
  for(int b=sizeof(int);b>0;b--){
    int count1=0,count2=0;
    int tmp1 = num1 >> ____(1)____ ;
    int tmp2 = num2 >> ____(1)____ ;
    while (tmp1 > 0) {
      if ( ____(2)____ )
        count1++;
      tmp1 = tmp1 >> ____(3)____ ;
    }
    while (tmp2 > 0) {
       if ( ____(4)____ )
         count2++;
       tmp2 = tmp2 >> ____(3)____ ;
    }
    if ( ____(5)____ )
      return 0;
  }
 return 1;
}
int main (){
  int num1, num2, status;
  scanf("%d %d",&num1,&num2);
  status=compare_numbers(num1,num2);
  (status==1?printf("Matches\n"):printf("Does not match\n"));
  return 0;
}
```

```
******** Answer ********
(1) (b-1)*8
(2) tmp1 & 1
(3) 1
(4) tmp2 & 1
(5) count1!=count2
```

```
******** Question ********
For converting an hexadecimal number into the equivalent binary number (32 bit r
epresentation), a C program is given below. Complete the program for getting the
 desired result. Note:  the ASCII value of '0' is 48 and of ‘A’ is 65.
```

```
int main()
{
  char hex[20], bin[33];  long n=0;  int i,k, val=0, len;
  printf("Enter any hexadecimal number: ");
  scanf("%s", hex);
  len = strlen(hex);  len--;
  for(k=0; hex[k]!='\0'; k++) {
    if(hex[k]>='0' && hex[k]<='9')
      val = hex[k] - ____(1)____ ;
    else if(hex[k]>='A' && hex[k]<='F')
      val = hex[k] - ____(2)____ ;
    n += val * pow( ____(3)____ );  // pow(x, y) gives x raised to y
    len--;
  }

  //convert decimal n to binary string
  i = 31;
  while(n && (i >=0))
```

```
  {
    if(____(4)___) {
      bin[i] = '1';
      n = ____(5)____ ;
    }
    else {
      bin[i] = '0';
      n = ____(5)____ ;
    }
    i--;
  }
  for( ; i>=0; i--)  bin[i]= '0';
  bin[32]='\0';
  printf("Binary is %s \n",bin);
  return 0;
}
```

******** Answer ********
(1) 48
(2) 55
(3) (16, len)
(4) n % 2
(5) n/2

******** Question ********
The following C program takes as inputs a positive integer (4bytes) and prints t
he binary in the following format-
<most significant bit> | <30th to 23rd bits> | <22nd to 0th bits>
For example,
Input: 5
Output: 0 | 00000000 | 00000000000000000000101
Input: 2147483647
Output: 0 | 11111111 | 11111111111111111111111

```
void printBinary(int n, int i)
{
    // Prints the binary representation of a number n up to i-bits.
    int k;
    for (k = i - 1; k >= ____2____ ;  k--) {
        if ((n >>  ____4____ ) & 1)
            printf("1");
        else
            printf("0");
    }
}

void main( )
{
    //integer scanned in int var
    printf("%d | ", var >> ____(1)____ );
    printBinary( var >> 23 , ____(3)____ );
    printf(" | ");
    printBinary(var , ____(5)____ );
    printf("\n");
}
```

******** Answer ********
(1) 31
(2) 0

(3) 8
(4) k
(5) 23

==========

==========
++++++++++++++++++++++ Topic-3: Structures and Pointers ++++++++++++++++++++++

******** Question ********
A structure stores the birthday information (year, month, and day).

```
struct DOB {
   int year, month, day;
};
```

Fill in the following blanks to inform whether a person took birth before India’s independence or after.

```
void ageCalc(struct DOB birthday)
{
    ____(1)____ ;
    iday.year=1947;
    iday.month=8;
    iday.day=15;
    int older=0;

    if( ____(2)____ ) {
        older=1;
    }
    else if(birthday.year==iday.year) {
        if( ____(3)____ )
            older=1;
        else if(birthday.month==iday.month) {
            if( ____(4)____ )
                older=1;
        }
    }
    ( ____(5)____ )? printf(“Born before Independence Day”): printf(“Born on or after Independence day”);
}
```

******** Answer ********
(1) struct DOB iday
(2) birthday.year < iday.year
(3) birthday.month < iday.month
(4) birthday.day < iday.day
(5) older == 1

******** Question ********
A structure stores the student academic information like, name, roll number, CGPA, number of credits passed.

```
struct Stud {
     char name[40];
      char roll[10];
      float CGPA;
```

```
     int credit;
};
```

Fill in the following blanks to report the name and roll number of the student w
ith highest CGPA in the pool of students who has cleared highest credits.

```
void studOfTheYear(struct Stud firstYr[], int num_of_stud)
{
  int i, maxCredit=0;
  int best=0.0;
  int top=-1;
  for(i=0; i<no_of_stud; i++) {
     if( ____(1)____ ) {
        maxCredit=firstYr[i].credit;
        ____(2)____ ;
        top=i;
     }
     else if( ____(3)____ ) {
        if( ____(4)____ )  {
           ____(5)____ ;
           top=i;
        }
     }
  }
  printf("Name: %s\tRoll: %s\n", firstYr[top].name, firstYr[top].roll);
}
```

```
******** Answer ********
(1) maxCredit < firstYr[i].credit
(2) best = firstYr[i].CGPA
(3) maxCredit == firstYr[i].credit
(4) best < firstYr[i].CGPA
(5) best = firstYr[i].CGPA
```

```
******** Question ********
Fill in the following blanks to count the number of words (separated only by sin
gle space)
from a sentence (array of characters) terminated by dot(.).
```

```
int count()
{
   int cnt=0;
   char *p,snt[1000];
   gets(snt);
   if(snt[0]!='\0') {
       p=snt;
       while( ____(1)____ ) {    // using pointer p variable
            if( ____(2)____ ) {   // using pointer p variable
               ____(3)____ ;
            }
            p++;
       }
       ____(4)____ ;
   }
   ____(5)____ ;
}
```

```
******** Answer ********
(1) *p!='\0'
```

(2) *p==’ ‘
(3) cnt++   Altr   cnt+=1   Alt   cnt=cnt+1
(4) cnt++   Altr   cnt+=1   Alt   cnt=cnt+1
(5) return cnt

******** Question ********
Complete the following program uses a function to take user inputs and then fill the same into a structure variable. Assume that all the strings involved have length at most 19.

```
struct student{
  char name[ ____(1)____ ];
  char roll[ ____(2)____ ];
};

____(3)____ init(void)
{
  struct student A;
  scanf("%s %s", A.name, A.roll);
  ____(4)____ ;
}
int main()
{
  struct student A;
  ____(5)____ ;   // Function call to fill the structure variable A by user inputs.
  return 0;
}
```

******** Answer ********
(1) 20 (any integer >= 20 to be treated as correct answer)
(2) 20 (any integer >= 20 to be treated as correct answer)
(3) struct student
(4) return A
(5) A=init()

******** Question ********
Fill in the blanks to complete the following function print_record() that prints the elements of an integer array of size 10. The blanks ____(3)____, ____(4)____ and ____(5)____ must be filled with expressions involving variables p and q only.

```
____(1)____  print_record(int A[10])
{
    ____(2)____ *p, *q;
    p = A;
    q = A+9;
    do {
        printf("%d ", ____(3)____ );
    } while( ____(4)____ != ____(5)____ );
    return;
}
```

******** Answer ********
(1) void
(2) int
(3) *p++
(4) p

(5) q+1

******** Question ********
Let A be an integer array of size 10, containing the numbers 1 to 10 in ascendin
g order. Let p and q be integer pointers initialized to A and &A[9] respectively
. What do the following expressions return?
(1) *(p+3)
(2) *(q-2)
(3) q-p
(4) (p+5)==(q-4)
(5) *++p

******** Answer ********
(1) 4
(2) 8
(3) 9
(4) 1
(5) 2

******** Question ********
You need to compute the roots of a quadratic equation ax^2 + bx + c = 0. The sol
ve() function mentioned below will return the number of roots as well as the val
ue of the root(s). You need to complete the blanks in order to achieve the objec
tive.
(a^b reads as, a to-the-power b)

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

int solve(double a, double b, double c, double *x1, double *x2) {
    if( ____(1)____ ) {
        *x1=(-b+sqrt(pow(b,2)-(4*a*c)))/(2*a);
        if( ____(2)____ ){
            ____(3)____ ;
            return 2;
        }
        return 1;
    }
    ____(4)____ ;
}

int main(void){
    double a, b, c;
    scanf("%lf%lf%lf", &a, &b, &c);
    double x1, x2;
    int count = solve( ____(5)____ );
    if(!count){
        printf("No real solution");}
    else if(count == 1){
        printf("The solution is x: %lf", x1);}
    else if(count>1){
        printf("The solutions are x1: %lf x2: %lf", x1, x2);}
}
```

******** Answer ********
(1) pow(b,2)-(4*a*c)>=0
(2) pow(b,2)-(4*a*c)>0
(3) *x2=(-b-sqrt(pow(b,2)-(4*a*c)))/(2*a)

(4) return 0
(5) a, b, c, &x1, &x2

******** Question ********
Given a string, the function "isPalindrome" attempts to check if the input string is a palindrome or not. A palindrome is a sequence of characters which reads the same backward as forward such as madam, radar etc.

```
void isPalindrome(char* string)
{
        char *ptr, *rev;
        ptr = string;

        while (*ptr != '\0') {
                ++ptr;
        }
        ____(1)____ ;

        for ( ____(2)____ ) {
            if ( ____(3)____ ) {
                --ptr;
                ____(4)____ ;
            }
            else
                break;
        }

        if ( ____(5)____ )
            printf("String is a Palindrome");
        else
            printf("String is NOT a Palindrome");
}
```

******** Answer ********
(1) --ptr
(2) rev = string; ptr >= rev;
(3) *ptr == *rev
(4) rev++
(5) rev > ptr

******** Question ********
The following code takes as input two complex numbers in the form of a structure. It performs the addition of these numbers. The structure for a complex number consists of two integers. Complete the following code.

```
#include<stdio.h>

struct complex {
    int real;
    int img;
};

int main() {
    struct complex a, b, sum, prod;
    printf("Enter the real and imaginary part of the first number : ");
    scanf("%d%d", ____(1)____ );
    printf("Enter real and imaginary part of the second number : ");
    scanf("%d%d", ____(2)____ );
```

```
    sum.real =  ____(3)____ ;
    sum.img = a.img + b.img;
    printf("Sum = %d + i%d",sum.real, sum.img);

    prod.real = ____(4)____ ;
    prod.img = ____(5)____ ;
    printf("Product = %d + i%d", prod.real, prod.img);
}
```

```
******** Answer ********
(1) &a.real, &a.img
(2) &b.real, &b.img
(3) a.real + b.real
(4) (a.real*b.real - a.img*b.img)
(5) (a.real*b.img + a.img*b.real)
```

```
******** Question ********
Consider a student structure, namely, student that contains the roll number (int
), name, marks in three subjects, and result which is either “FAIL” or “PASS”. T
he result is FAIL if all three subject marks are greater or equal to 30. The use
r takes input for n number of student records and prints their result by calling
 the function char *result (int mark[]). Fill up the blanks  in the below given
code to achieve the above stated objectives.

struct student{
  int roll;
  char name[50];
  int marks[3];
};

char *result (int mark[]);

void main(){
  int i, j, n;
  printf("Enter the number of students\n");
  scanf("%d",&n);

 //create suitable instance(s) of student structure with name std to hold n stud
ent records
  ____(1)____ ;

  for(i=0; i<n; i++){
    printf("Enter roll number and name of student %d:", i+1);
    scanf("%d %s", ____(2)____ );
    printf("Enter three subject marks of student %d:\n", i+1);
    for (j=0; j<3; j++)
     scanf("%d", ____(3)____ );
  }
  for(i=0; i<n;i++){
    // calling the result function to compute result
    printf("result status of student %d is:%s\n", i+1, ____(4)____ );
  }
}

char *result (int mark[]);
{
    // Logic for pass and fail
    if( ____(5)____ )
```

```
       return "PASS";
    else
       return "FAIL";
}
```

```
******** Answer ********
(1) struct student std[n]
(2) &std[i].roll, std[i].name
(3) &std[i].marks[j]
(4) result(std[i].marks)
(5) mark[0]>=30 && mark[1]>=30 && mark[2] >=30
```

******** Question ********
Given two arrays, namely, SrcArray and DstArray of the same size stored in variable size. The function void SWAP(int * SrcArray, int * DstArray, int size) swaps the elements of two arrays, i.e. after the swap function, SrcArray contains the elements of the DstArray and vice versa. Fill the blanks based on the comments written in the code snippet given below to get desired output.

```
void SWAP(int * SrcArray, int * DstArray, int size)
{
    // Pointer to the last element of the source array
    int * SrcEnd = ____(1)____ ;

    // Pointer to the last element of the destination array
    int * DstEnd   = ____(2)____ ;
    int temp=0;

    // Swap elements of both the arrays here

    while(SrcArray <= SrcEnd && DstArray <= DstEnd) {
        temp = *SrcArray ;
        // Assigning destination array value to the source array
        ____(3)____ ;

        *DstArray   = temp;
        // Increment the pointer in source array to point to the next element
        ____(4)____ ;

        // Increment the pointer in destination array to point to next element
        ____(5)____ ;
    }
}
```

```
******** Answer ********
(1) SrcArray + (size - 1)
(2) DstArray + (size - 1)
(3) *SrcArray = *DstArray
(4) SrcArray++
(5) DstArray++
```

******** Question ********
Consider two strings: string1 and string2. The function void compare(char *string1, char * string2) checks whether both the strings are equal or not. Another function void concat(char *string1, char * string2) appends string2 after string1. Code snippets with some statements missing are given below. Fillup the blanks to get the desired output.

```
void compare(char * string1, char * string2)
{
    int diff;
    //check elements are equal, none of the string pointers have reached their e
nds
    while( ( ____(1)____ ) && (*string1 == *string2) )
    { string1++;  string2++;
    }
   //check the difference between two string lengths

    diff= ____(2)____ ;
        if(diff==0)
            printf("Both strings are equal.");
    else
            printf("Both strings are different.");
}

void concat(char *string1, char * string2)
{
    char * s1 = string1;
    char * s2 = string2;
    // Move till the end of string1 using s1 pointer
    while(*( ____(3)____ ));
    --s1;
    // Append string2 at the end of string 1 using pointer s2
    while(*(s1++) = *( ____(4)____ ));
    //Print the string after concatenation
    printf("Concatenated string is:%s \n", ____(5)____ );
}
```

```
******** Answer ********
(1) *string1 !='\0' && *string2 !='\0'
Alt. Ans.:  *string1 && *string2
(2) *string1 - *string2
Alt. Ans.:  *string2 - *string1
(3) s1++
(4) s2++
(5) string1
```

==========

==========

+++++++ Topic-4: Multi-Dimensional Arrays and Dynamic Memory Allocation ++++++++

```
******** Question ********
Fill in the following blanks to read an image (in the form of a matrix) and
will return the maximum intensity values (matrix element) of the image.

int maxIntensity(int row,int col)
{
   int **image, i,j,maxI=0;
   image = ____(1)____ ;
   for(i=0;i<row;i++)
       image[i] = ____(2)____ ;
   ReadImage(image,row,col);   //function that will read intensity
   for(i=0;i<row;i++)
     for(j=0;j<col;j++)
        if(maxI<image[i][j])
```

```
            maxI=image[i][j];
   for(i=0; i< ____(3)____ ; i++)
       free( ____(4)____ );
   free( ____(5)____ );
   return maxI;
}
```

```
******** Answer ********
(1) (int **)malloc(sizeof(int *)*row)
(2) (int *)malloc(sizeof(int)*col)
(3) row
(4) image[i]
(5) image
```

```
******** Question ********
Fill in the following blanks to multiply two matrices y and z and store the resu
lt in x.

void multiply (int *matA, int *matB, int *matC, int N)
{
   int i, j, k,offset,offset1,offset2;
   for (i=0; i<N; i++) {
      for (j=0; j<N; j++) {
         offset = ____(1)____ ;
         *(matA +offset) = 0.0;
         for (k=0; k<N; k++) {
             offset1 = ____(2)____ ;
             offset2 = ____(3)____ ;
             *(matA + ____(4)____ ) += (*(matB +offset1))* (*(matC + ____(5)____
 ));
         }
      }
   }
}
```

```
******** Answer ********
(1) N*i + j
(2) N*i + k
(3) N*k + j
(4) offset
(5) offset2
```

```
******** Question ********
Fill in the following blanks to store the first name of 10 students in an array
whose length may vary.

void readStore()
{
    char ____(1)____ ;
    int i, n, j;
    namelist = ____(2)____ ;
    for(i=0,n=0; i<10; i++) {
        scanf("%s", fname);
        n = 0;
        while(fname[n++] != '\0');
        namelist[i] = ____(3)____ ;
        for(j=0; j<n; j++) {
            namelist[i][j] = fname[j];
```

```
        }
    }
    for(i=0; i<10; i++) {
        printf(“%s\n”, namelist[i]);
    }
    for(i=0; i<10; i++)
        free( ____(4)____ );
    free( ____(5)____ );
}
```

******** Answer ********
(1) **namelist,fname[100]
(2) (char **)malloc(sizeof(char *)*10)
(3) (char *)malloc(sizeof(char)*n)
(4) namelist[i]
(5) namelist

******** Question ********
Consider the following structure type.

```
struct student
{
  char roll[20];
  char name[20];
}
```

Fill in the blanks to complete the following function to
(A) dynamically allocate an array of n structures and assign the base address to a pointer p of a type suitable to traverse the array by pointer arithmetic, and
(B) print the strings in the name field of the structure variables in the array.

```
void myfill()
{
  int i;
  ____(1)____ *p;
  scanf("%d", &n);
  p = ____(2)____ malloc( ____(3)____ * ____(4)____ );
  for(i=0; i<n; )
    printf("%s\n", ____(5)____ );
}
```

******** Answer ********
(1) struct student
(2) (struct student*)
(3) n
(4) sizeof(struct student)
(5) p[i++].name

******** Question ********
Consider the following declarations:

```
int A[2][3];
int B[10];
```

Write the data types of
(1) &A[1][3]
(2) B
(3) A[1]

(4) *(A+2)
(5) *(*(A+3)+4)

******** Answer ********
(1) int *
(2) int *
(3) int *
(4) int *
(5) int

******** Question ********
Fill in the blanks to complete the following program that creates a dynamic array of structure pointers whose entries point to dynamic arrays of structures containing information of students.

```
struct student
{
  char roll[20];
  char name[20];
};

int main()
{
  int i, j, n;
  struct student **list;
  int *size;
  scanf("%d", &n);  // scans the number of lists of student records
  list = ( ____(1)____)malloc( ____(2)____ * sizeof(struct node*));
  size = (int*)malloc(n*sizeof(int)); //dynamic array to store the sizes of various lists
  for(i=0; i<n; i++)
  {
    scanf("%d ", &size[i]); //scans size of i-th list
    list[i] = ( ____(3)____ )malloc( ____(4)____ ));
    for(j=0; ____(5)____; j++)
      scanf("%s %s", list[i][j].name, list[i][j].roll);
  }
  return 0;
}
```

******** Answer ********
(1) struct student**
(2) n
(3) struct student*
(4) size[i]*sizeof(struct node)
(5) j<size[i]

******** Question ********
The below program takes a matrix of size m x n and traverses the matrix in spiral form. The function spirallyTraverse() takes the matrix, m and n as input parameters and prints the integers in order that denote the spiral traversal of the matrix.

Test Case 1:
Input:
3 3
1 2 3 4 5 6 7 8 9

Test Case 2:
Input:
4 3
1 2 3 4 5 6 7 8 9 10 11 12

```
Output:                              Output:
1 2 3 6 9 8 7 4 5                    1 2 3 6 9 12 11 10 7 4 5 8

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

void spirallyTraverse(int **arr,int m, int n)
{
     //k = start row index
    //l = start col index

    int k=0;
    int l=0;
    int i;

    while( ____(2)____ )
    {
        for(i=l;i<n;i++)
            printf("%d ",  *(*(arr+k)+i));
        k++;
        for(int i=k;i<m;i++)
            printf("%d ", ____(3)____ );
        n--;
        if(k<m)
        {
            for(int i=n-1;i>=l;i--)
                printf("%d ", ____(4)____ );
            m--;
        }
        if(l<n)
        {
            for(int i=m-1;i>=k;i--)
                printf("%d ", ____(5)____ );
            l++;
        }
    }
    printf("\n");
}

int main(void)
{
    int m,n;
    scanf("%d %d",&m,&n);
    //Allocates the memory dynamically
    int **arr = (int **)malloc(m * sizeof(int *));
    *arr =  ____(1)____ ;
    for (int i = 0; i < m; i++)
        arr[i] = (*arr + n*i);

    for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
            scanf("%d",&arr[i][j]);

        spirallyTraverse(arr,m,n);
    return 0;
}

******** Answer ********
(1) (int *)malloc(sizeof(int)*m*n)
(2) k<m && l<n
```

(3) *(*(arr+i)+(n-1))
Alt. Ans.: arr[i][n-1])
(4) *(*(arr+(m-1))+i)
Alt. Ans.: arr[m-1][i])
(5) *(*(arr+i)+l)
Alt. Ans.: arr[i][l])

******** Question ********
Given an array, the task is to form a spiral matrix.
Input:
arr[] = { 1, 2, 3, 4, 5,
          6, 7, 8, 9, 10,
          11, 12, 13, 14, 15, 16 };

Output:
  1 2 3 4
 12 13 14 5
 11 16 15 6
 10 9 8 7

Given the code to create the spiral matrix, fill in the blanks:

```
#include <stdio.h>

#define R 3
#define C 6

void formSpiralMatrix(int arr[], int mat[R][C])
{
    int top = 0, bottom = R - 1, left = 0, right = C - 1;
    int index = 0;

    while (1) {
        if ( ____(1)____ )
            break;

        // top row
        for (int i = left; i <= right; i++)
            mat[top][i] = arr[index++];
        top++;

        if ( ____(2)____ )
            break;

        // right column
        for (int i = top; i <= bottom; i++)
            mat[i][right] = arr[index++];
        right--;

        if ( ____(3)____ )
            break;

        // bottom row
        for (int i = right; i >= left; i--)
            mat[bottom][i] = arr[index++];
        bottom--;

        if ( ____(4)____ )
            break;
```

```
        // left column
        for (int i = ____(5)____ ; i >= top; i--)
            mat[i][left] = arr[index++];
        left++;
    }
}

void printSpiralMatrix(int mat[R][C])
{
    for (int i = 0; i < R; i++) {
        for (int j = 0; j < C; j++)
            printf("%d ",mat[i][j]);
        printf("\n");
    }
}

int main()
{
    int arr[]
        = { 1, 2, 3, 4, 5, 6,
            7, 8, 9, 10, 11, 12,
            13, 14, 15, 16, 17, 18 };
    int mat[R][C];

    formSpiralMatrix(arr, mat);
    printSpiralMatrix(mat);

    return 0;
}
```

```
******** Answer ********
(1) left > right
(2) top > bottom
(3) left > right
(4) top > bottom
(5) bottom
```

```
******** Question ********
Let’s write a C program to read and print the details of N students using struct
ure and Dynamic Memory Allocation.
```

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

#define mem_size 80

/*structure declaration*/
typedef struct
{
    char* firstName;
    char* lastName;
    int roll;
    float perc;
}student;

student pstd[100];

int main()
```

```
{
    int n,i;

    printf("Enter total number of elements: ");
    scanf("%d",&n);

    /*read and print details*/
    for(i=0; i<n; i++)
    {
        printf("\nEnter detail of student [%3d]:\n",i+1);
        pstd[i].firstName = ____(1)____ ;   //allocate memory = mem_size for fir
st name
        pstd[i].lastName = ____(1)____ ;    //allocate memory = mem_size for las
t name
        printf("Enter first name: ");
        scanf(" "); /*clear input buffer*/
        scanf("%s", ____(2)____ );
        printf("Enter last name: ");
        scanf("%s", ____(3)____ );
        printf("Enter roll number: ");
        scanf("%d", ____(4)____ );
        printf("Enter percentage: ");
        scanf("%f",&(pstd[i].perc));
    }

    printf("\nEntered details are:\n");
    for(i=0; i<n; i++)
    {
        printf("%20s \t %20s \t %5d \t %.2f\n", ____(5)____ );
    }

    for (i=0; i<n; i++)
    {
        free(pstd[i].firstName);
        free(pstd[i].lastName);
    }
    return 0;
}

******** Answer ********
(1) malloc(mem_size)
(2) pstd[i].firstName
(3) pstd[i].lastName
(4) &(pstd[i].roll)
(5) pstd[i].firstName, pstd[i].lastName, pstd[i].roll, pstd[i].perc


******** Question ********
Let us consider a square matrix A of size m x m. A is said to be an upper triang
ular matrix iff A[i[j]=0 for every i>j. For large values of m, it does not make
sense to store all values of the matrix A, since almost half the elements of A a
re 0.  Using dynamic memory allocation, one can ensure that only non-zero elemen
ts of these matrices are stored.  Consider the following upper triangular matrix
 of size 4x4 stored with zeros in memory.

row 0:  1 2 4 7
row 1:  0 3 5 8
row 2:  0 0 6 9
row 3:  0 0 0 10
```

Using the optimized storage memory layout, this can be done as follows.

```
row 0:  1 2 4 7
row 1:  3 5 8
row 2:  6 9
row 3:  10
```

Consider the following incomplete function int **convert(int **mat, int m) which takes as input a 2D matrix mat of size mxm , creates an upper triangular matrix upper with minimum memory dynamically, initializes it using values from mat and finally returns it. Fill in the required blanks marked by (1)-(5).

```
int **convert(int **mat,int m)
{
    int **upper;
    int i,j;
    //Code for dynamically allocating
    upper =(int **)malloc( ____(1)____ * sizeof( ____(2)____ ));
    for(i=0;i<m;i++)
        upper[i]=(int*)malloc(( ____(3)____ * sizeof(int));
    //Code for populating
    for(i=0;i<m;i++)
        for(j=0;j<m;j++) {
            if( ____(4)____ ) {
                ____(5)____ = mat[i][j];
            }
        }
   return upper;
}
```

******** Answer ********

(1) m
(2) int*
(3) (m-i)
(4) (i<=j)
(5) upper[i][j-i]

******** Question ********

Let us consider a square matrix A of mXm size which we represent as a 2D array A[N][N]. We want to rotate this matrix 90 degrees anti-clockwise and print it. This can be done by first applying a transpose operation where elements of rows become elements of columns. After this, the columns of the transposed matrix are reversed. For example,

```
Row 0: 1  2  3                Row 0: 1  4  7              Row 0: 3  6  9
Row 1: 4  5  6    Transpose   Row 1: 2  5  8    Reverse   Row 1: 2  5  8
Row 2: 7  8  9      --->      Row 2: 3  6  9     --->     Row 2: 1  4  7
```

Let us consider the following incomplete program which
(i) takes the size and integer elements of the matrix as inputs from the user,
(ii) rotates the matrix by 90 degrees anti-clockwise.
(iii) prints the rotated matrix before exiting.

Fill in the blanks in the following code.

```
int main()
{
int N,i,j,k;
printf("enter number of rows or columns in square matrix:");
scanf("%d",&N);
```

```
int a[N][N];
for (i = 0; i < N; i++) {
  printf("enter row wise:\n");
  printf("\trow %d elements:\n",i);
  for (j = 0; j < N; j++){
    printf("\t\tat (row %d, column %d):\t",i,j);
    scanf("%d", &a[i][j]);
 }
}

// transpose the matrix
for (i = 0; i < N; i++) {
  for (j = i; ____(1)____ ; j++) {
    int tmp = a[i][j];
    a[i][j] = ____(2)____ ;
    ____(2)____ = tmp;
  }
}
// reverse columns to turn the matrix upside down
for (i = 0; i < N; i++) {
  for (j=0,k=N-1; ____(3)____ ; j++, k--) {
    int tmp = ____(4)____ ;
    ____(4)____ = ____(5)____ ;
    ____(5)____ = tmp;
  }
}

// Display square matrix after 90 degrees left rotation
for (i = 0; i < N; i++) {
  for (j = 0; j < N; j++)
    printf("%d ", a[i][j]);
  printf("\n");
}
return 0;
}
```

```
******** Answer ********
(1)  j < N
(2)  a[j][i];
(3)  j<k
(4)  a[j][i]
(5)  a[k][i]
```

```
******** Question ********
Let us consider a 1D array B of size k. We want to convert this 1D array into a
new 2D array A of m x n dimensions such that k=m*n.  Consider a function make2D(
int *B,int m,int n)  which performs this and returns the resultant 2D array A. T
he function copies consecutive n elements of B and stores in one row of A. This
is done m times. See the following example to understand how it converts the 1D
array into 2D array.

If the array, B[6] = {1, 2, 3, 4, 5, 6},  then
make2D(B,2,3) returns A = { {1, 2, 3}, {4, 5, 6} }

Here we want to convert an 1D array B (with 6 elements) into a 2D array A, that
has 2 rows, each with 3 elements (3 columns). So, the first 3 elements of B are
stored in the first row of A. The next 3 elements of B are stored in the second
row. Consider the following incomplete code snippet for the aforementioned funct
ion. Fill in the required blanks marked by the roman numerals (i) - (v).
```

```
int**  make2D(int *b, int m, int n) {
  int** a = (int**)malloc( ____(1)____ );
  for(int i = 0; i < m; i++) {
    ____(2)____ = (int*) malloc( ____(3)____ );
  }
  for(int i = 0; i < m; i++) {
    for(int j = 0; j < n; j++) {
      ( ____(4)____ ) = ( b[ ____(5)____ ] );
    }
  }
  return a;
}
```

******** Answer ********
(1) m*sizeof(int*)
(2) a[i]
(3) n*sizeof(int)
(4) a[i][j]
(5) n*i+j

==========