Q3. Question based on Data Structure(Array/Stack/Queues) Total Marks 12-14
Address Calculation 3 Marks/ Postfix-Infix 2 Marks/ Stack-Queues 4 Marks /Arrays 3-5

Address Calculation

Row Major Order :

Address of A[R][C]=BA+W[(R-LBR)*N+(C-LBC)]

Column Major Order :

Address of A[R][C]=BA+W[(I-LBR)+(J-LBC)*M]

Where : BA = Base Address W=Size of elements I,J=Find or determine the location LBR=Lower Bound of row N= Number of columns LBC=Lower bound of column

- **LBR & LBC=0** and UBR =N and UBC=M : if 2- DEMENSIONAL ARRAY IS IN THE FORM OF **A[N][M]**
  FOR EXAMPLE -> **A[20][30]**

- BUT IF 2- DEMENSIONAL ARRAY IS IN THE FORM OF **A[LBR……..N][LBC……LBCM]** FOR EXAMPLE -> **A[2……..20][4……..30]**

  M=UBC-LBC+1; N=UBR - LBR+1;

Solved Examples

A double dimension array DA[20][40] is stored in computer's main storage. Every element of DA requires 8 bytes of memory. If the base address of DA is 1000 then calculate the address of DA[20][30] when the array is stored as: i)Row Major ii)Column Major

N = 20, M = 40, BA = 2000, W = 8, R = 20, C = 30 LBC=0,LBR=0

Row Major: Address of DA[R][C] = BA + W * (R-LBR) * M + (C-LBC))

Address of DA[20][30]= 2000 + 8 * ((20-0) * 40 + (30-0)) = **8640**

Col Major: Address of DA[r][c] = BA + W * (R-LBR) + (C-LBC)*N)

Address of DA[20][30]= 2000 + 8 * ((20-0) + (30-0)*20)= **6960**

1. An array x[8][20] is stored in the memory with each element requiring 2 bytes of storage. If the base address of the array is 2500, calculate the location of x[5][5] when the array x is stored using the column major order and row major order.
2. An array Arr[1..20][1..20] is stored in the memory with each element requiring 4 bytes of storage. If the base address of array Arr is 2000, determine the location of Arr[15][9] when the array Arr is stored in (1) Row wise and (2) Column wise.
3. An array MAT[30][10] is stored in the memory row wise with each element occupying 8 bytes of memory. Find out the base address and the address of the element MAT[15][5], if the location of MAT[5][7] is stored at the address 3000.
4. An array MAT[20][25] is stored in the memory with each element requiring 2 bytes of storage. If the base address of MAT is 4000 MAT[[12][8] when the array stored in (i) RMO and (ii) CMO
5. An array ARR[15][20] is stored in the memory, along the row with each element occupying 4 bytes . Find out the base address and the address of the element ARR[3][2] if the element ARR[5][2] is stored at the address 1500.
6. Each element of an array Data[20][50] required 4 bytes of storage, Base address of data is 2000, determine the location of Data[10][10] when the array is stored as (a) row major (b) column major
7. An array A[11][21] is stored in the memory with each element requiring 2 bytes of storage. If the base address of array in memory is 300, determine the location of a[5][31] when the array is A stored by (i) Row major (ii) Column major
8. An array VAL[1....15][1...10] is stored in the memory with each element requiring 4 bytes of storage. If the base address of array VAL is 1500, determine the location of VAL[12][9] when the array VAL is stored (i)row-wise (ii)column-wise
9. An array ARR[15][35] is stored in the memory along the column with each of its elements occupying 8 bytes. Find out the bas address and the address of an element Arr[2][5], if the location Arr[5][10] is stored at the address 4000.
10. An array Arr[35][15] is stored in the memory along the row with each of its elements occupying 4 bytes. Find the bas address and the address of an element Arr[20][5], if the location Arr[2][2] is stored at the address 3000.
11. *An array arr[15][20] is stored in the memory along the row with each element occupying 4 bytes. Find out the base address and address of the element arr[3][2], if the element arr[5][2] is stored at the address 1500.*
12. *An array MAT[30][10] is stored in the memory column wise with each element occupying 8 bytes of memory. Find out the bas address and the address of element MAT[5][7] is stored at the address 1000.*
13. If an array B[11][8] is stored as column wise and B[2][2] is stored at 1024 and B[3][3] at 1084, find the address of B[5][3] and B[1][1].
14. An array Arr[50[100] is stored in the memory along the row with each element occupying 2 bytes. Find out the address of the location ARR[20][50] if location of Arr[20][30] is 1350.
15. An array x[30][10] is stored in the memory with each element requiring 4 bytes of storage. If the base address of x is 4500, find out memory locations of x[12][8] and x[2][4], if the content is stored along the row.
16. An array ARR[15][35] is stored in the memory along the column with each of its elements occupying 8 bytes. Find out the base address and the address of an element ARR[2][5] , if the location is stored at the address 4000
17. An array X[15][10] is stored in memory with each element requiring 2 bytes of storage. If the base address of array is 2000, calculate the location of X [7][8] when the array is stored by (1) row major order (2) column major order.

18. X [1..6][1….10] is a two dimensional array. The first element of the array is stored at Location 100. Each element of the array occupies 6 bytes. Find the memory location of X[2][4] when (i) array is stored row wise. (ii)array is stored column wise
19. Each element of an array A[-20..20,10..35] requires one byte of storage. If the array is stored in column major order beginning location 500, determine the location of A[0,30].
20. An array S[35][15] is stored in the memory along the row with each of its elements occupying 4 bytes. Find out the memory location for the element S[20][5], if an element S[2][2] is stored at the memory location 3000.
21. Given the two dimensional array a[10][20] base address of a is 100 and width of each element is 4 bytes. Find the location of a[8][15] when the array is stored as column-wise and row-wise
22. An array A[-2..8][-2..5] is stored in the memory along the column with each element occupying 4 bytes. Find out the address of the element A[3][2].

QUESTION BASED ON POSTFIX

Q1- Convert the following expression to Postfix expression

(i) (A+B)*C+D/E-F
(ii) A+B*C^D-(E/F-G)
(iii) A-B+C^D*G/H (Evaluate the postfix expression showing the status of stack at every step gives A=5,B=2, C=4,D=2,E=1,G=6,H=5
(iv) A+(B*C-(D/E^ F)*G)*H(A+ B)*(C*D-E)*F / G
(v) (A + B ^ D)/(E-F)+G
(vi) A*(B+D)/ E-F*(G +H/K)
(vii) (a + b)* (c + d)/(e + f)^ g
(viii) (m+ n)*(k + p)/(g / b) ^(a^ b / c)
(ix) Q = [(A + B)/(C + D)^ (E / F)]+ (G + H)/ I
(x) A*(B+ D)/ E - F(G +H/ k)
(xi) P = - A + B * C ^ (D * E) / F
(xii)

Q-2 Evaluate the following Postfix notation of expression, show status of stack for each operation :

(i) 500,20,30,+,10,*,+ Ans :1000
(ii) 120,45,20,+,25,15,-,+,* Ans :9000
(iii) 20,45,+,20,10,-,15,+,* Ans :1625
(iv) 100,40,8,+,20,10,-,+,* Ans :5800
(v) 5,11,-,6,8,+,12,*,/ Ans :-1/28
(vi) 5,6,9,+,80,5,*,-,/ Ans :-1/77
(vii) 7,2,-,1,14,-,1,2,*,+,-
(viii) 5,3,2,4,+,5*,+6,+,- Ans :34
(ix) TRUE, FALSE, TRUE,FALSE, NOT,OR, TRUE, OR, OR, AND
(x) TRUE, FALSE AND, TRUE, TRUE, NOT, OR, AND
(xi) TRUE,FALSE,TRUE,NOT FALSE,TRUE OR,NOT,AND, OR,AND
(xii) (TRUE && FALSE) || !(FALSE||TRUE)

Array

1. Assume an array E containing elements of structure Employee is required to be arranged in descending order of Salary. Write a C++ function to arrange the same with the help of bubble sort, the array and its size is required to be passed as parameters to the function. Definition of structure Employee is as follows:

```
struct Employee
  {
    int Eno;
    char Name [25];
    float Salary;
  };
```

2. Write a C++ function revdup(int x[ ], int n) to remove the duplicate occurrence of the value present in an integer array.
For e.g.If array initially x[ ] = { 1,1,1,7,2,2,6 }
After removing duplicate values the array will be x[ ] = { 1,7,2,6 }

3. Given two arrays of integers X and Y of sizes m and n respectively. Write a function named MERGE( ) which will produce a third array named Z, such that the following sequence is followed. (i) All odd numbers of X from left to right are copied into Z from left to right.
(ii) All even numbers of X from left to right are copied into Z from right to left.
(iii) All odd numbers of Y from left to right are copied into Z from left to right. (iv) All even numbers of Y from left to right are copied into Z from right to left.

4. A 2-D array defined as A[4..7, -1..3] requires 2 words of storage space for each element. If the array is stored in row major form, calculate the address of A[6,2] given the base address is 100.

5. Write a function in C++ which accepts an integer array and its size as arguments and arrange all the odd numbers in the first row and even numbers in the second row of a two dimensional array. The unused cells of two dimensional array must be filled with 0.
If the array is 1, 2, 3, 4, 5, 6
The resultant 2-D array is given below

1	3	5	0	0	0
0	0	0	6	4	2

6. Define a function swapcol( ) in C++ to swap the first column elements with the last column elements, for a two dimensional integer array passed as the argument to the function.

7. An array S[40][30] is stored in the memory along the row with each of the element occupying 2 bytes, find out the memory location for the element S[20][10], if the Base Address of the array is 5000.

8. Write a user defined function named upper_half() which takes a two dimensional array A, with size N rows and N columns as argument and print the upperhalf of the array.

9. Write a function in C++ which accepts a 2-D array of integers and its size as arguments and prints no of even numbers and odd numbers in each column.
If the array is

11	12	31	41
52	62	71	82
9	10	11	12

The output will be

Column 1: Even numbers : 1 Odd numbers :
Column 2: Even numbers : 3 Odd numbers :
Column 3: Even numbers : 0 Odd numbers :
Column 4: Even numbers : 2 Odd numbers :

10. The numbers (89, 20, 31, 56, 64, 48) are required to be sorted using selection, insertion and bubble sort. Show how the list would appear at the end of each pass.

11. Write a program in C++ to use a function for replace the repeating elements in an array by 0 . The order of the array should not change.

Eg : Array : 10 , 20 , 10 , 50 , 30 , 20 , 10
Result : 10 , 20 , 0 , 50 , 30 , 0 , 0

12. Define a function swaparray(int [ ],int) that would accept a one dimensional integer array numbers and its size n. The function should rearrange the array in such a way that the values of alternate locations of the array are exchanged (assume the size of the array to be even)
Example:
If the array initially contains **2, 5, 9, 14, 17, 8, 19, 16**
The after rearrangement the array should contain **5, 2, 14, 9, 8, 17, 16, 19**

13. Write a function in C++ which accepts an integer array and its size as an arguments and exchanges first half side elements with second side elements of the array
Example:
If the given array as eight elements are
10, 15, 6, 16, 21, 20, 1, 0
The function should rearrange the array as
21, 20, 1, 0, 10, 15, 6, 16

Stack & Queues

Solved

Q1. Write a function in C++ to delete a node containing customer's information, from a dynamically allocated Queue of Customers implemented with the help of the following structure:

```
struct Customer
{       int CNo;
        char CName[20];
        Customer *Link;
};
```

Ans:

```
struct Customer
{       int CNo;
        char CName[20];
        Customer *Link;
} *Front, *Rear, *ptr;
void DELETE()
{       if(Front = = NULL)
        cout<<"\n Queue Underflow\n";
    else
    {   ptr = Front;
        Front = Front→Link;
        delete ptr;
    }
}
```

Q2. Write a function in C++ to delete a node containing Book's information, from a dynamically allocated Stack of Books implemented with the help of the following structure.

```
struct Book
{       int BNo;
        char BName[20];
        Book *Next;
      };
```

Ans:
```
struct Book
{
    int BNo;
    char BName[20]; Book
    *Next;
}*top, *ptr;
void POP()
{
if(top = = NULL)
    cout<<"\n Stack Underflow\n";
     else
  {
    ptr = top;
    top = top→Next;
    delete ptr;     }
 }
```

1. Define member function QInsert( ) to insert and QDel( ) to delete nodes of a linked list implemented class Queue having the following Definitions:

```
Struct Node
  {       char name[20];
          int age;
        Node *Link;
  };
class Queue
  {       Node *Rear, *Front;
          public:
          Queue( ) { Rear=NULL; Front = NULL}
          void QInsert( );
          void QDel( );
  };
```

2. Write a function in C++ to delete a node containing names of student, from a dynamically allocated stack of names implemented with the help of following structure :

```
struct student
{       char name[20];
        student *next;
};
```

3. Write a user defined function in C++ to insert an element from a dynamically allocated Queue where each node contains the long integer (schoolno) as data. Assume the following definition of SCHOOL for the same.

```
struct SCHOOL
{       long scno;
        SCHOOL * link;
};
```

4. Given the following class:

```
char *msg[]={"OVER FLOW","UNDER FLOW"};
class stack
{       int top;
        stk[5];
        void err_rep(int e_num)
        {cout<<msg[e_num];}          // report error message
          public:
                void init() {top=0;}
                 void push(int);              // put the new value in to stk
                void pop();                   // get the top valued from stk
};
```

Define pop outside the stack. In your definition take care of underflow condition. Function pop has to invoke err_rep to report over flow.

5. Consider the following portion of a program, which is implemented a linked list of library. Write the definition of function PUSH( ), to insert a new node in the stack with required information:

```
struct Library
{       int id;
        char name[20];
        Library *Link;
};
```

6. Write a function in C++ to delete a node containing names of student, from a dynamically allocated stack of names. The function receives the value of top by reference. The stack is implemented with the help of following structure :

```
struct student
{
        char name[20];
        student *next;
};
```

1. Write the definition of a member function PUSH( ) in C++, to add a new book in a dynamic stack of BOOKS considering the following code is already included in the program

```
struct BOOKS
{   char ISBN[20],  TITLE[80]; BOOKS *Link;
};
class STACK
{    BOOKS *Top;
    public:
    STACK(){Top=NULL;}
    void PUSH();
    void POP();
~STACK();
 }
```