



Internal Assessment Test I-April 2023													
USN											EC		
Branch:									Sub Code: 18EC643				
Subject:	Data Structure using C++							Sub Code: 18EC643					
Date:	25-04-2023	Duration:	90 Min's	Sem/Sec:	VI Sem-A/B/C/D								
S.N	Note : Answer any Five Full Questions										Marks	CO	RBT
1	Explain how dynamic memory allocation is performed in C++ with suitable program code.										10	CO1	L2
2a	Define Linked list. Explain the linked list representation. Write a pseudo code for node representation.										5	CO1	L2
2b	Write a C++ program to add two 2D array.										5	CO1	L2
3	What is sparse matrix? Explain in detail different ways in which sparse matrix is represented.										10	CO1	L2
4a	Define an Array. Write C++ program to store elements and retrieve elements from an array.										5	CO1	L1
4b	Write C++ program to find factorial of a given number and Fibonacci series using function.										10	CO1	L3
5a	Define template. Explain the syntax of template with suitable example.										5	CO1	L2
5b	Define function. Bring out the difference between local and global variables used in function.										5	CO1	L2
6	Write a C++ program with the help of template to input 2 numbers and output smallest number using class and object.										10	CO1	L3

Course Instructor

CCI signature

HoD Signature

Internal Assessment Test I-April 2023(SCHEME OF EVALUATION)												
USN										EC		
Branch:												
Subject:	Data Structure using C++						Sub Code:	18EC643				
Date:	25-04-2023		Duration:	90 Min's		Sem/Sec:	VI Sem-A/B/C/D					
S.N	Note : Answer any Five Full Questions									Marks	CO	RBT
1	<p>Explain how dynamic memory allocation is performed in C++ with suitable program code.</p> <p>The operating system uses dynamic memory allocation in C++ for dynamically allocated variables, for example, <code>int* ptr = new int;</code>, <code>int* arr = new int[6];</code>. Dynamically allocated memory does not get de-allocated until the program terminates. So, a programmer must de-allocate the memory, when it is no longer required.-2M</p> <p>The program will show the use of new and delete</p> <pre>#include <iostream> using namespace std; int main () { // Pointer initialization to null int* m = NULL; // Request memory for the variable // using new operator m = new(nothrow) int; if (!m) cout<< "allocation of memory failed\n"; else { // Store value at allocated address *m=29; cout<< "Value of m: " << *m <<endl; } // Request block of memory // using new operator float *f = new float(75.25); cout<< "Value of f: " << *f <<endl; // Request block of memory of size int size = 5; int *arr = new(nothrow) int[size]; if (!arr) cout<< "allocation of memory failed\n"; else { for (int i = 0; i< size; i++) arr[i] = i+1; cout<< "Value store in block of memory: "; for (int i = 0; i< size; i++)</pre>									10	CO1	L2

	<pre> cout<<arr[i] << " "; } // freed the allocated memory delete m; delete f; // freed the block of allocated memory delete[] arr; return 0; } -8M </pre>			
2a	<p>Define Linked list. Explain the linked list representation. Write a pseudo code for node representation.</p> <p>Like arrays, a Linked List is a linear data structure. Unlike arrays, linked list elements are not stored at a contiguous location; the elements are linked using pointers. They include a series of connected nodes. Here, each node stores the data and the address of the next node.2M</p> <pre> // A linked list node- 3M struct Node { int data; struct Node* next; }; </pre>	5	CO1	L2
2b	<p>Write a C++ program to add two 2D array.-5M</p> <pre> #include <stdio.h> int main() { int r, c, a[100][100], b[100][100], sum[100][100], i, j; printf("Enter the number of rows (between 1 and 100): "); scanf("%d", &r); printf("Enter the number of columns (between 1 and 100): "); scanf("%d", &c); printf("\nEnter elements of 1st matrix:\n"); for (i = 0; i < r; ++i) for (j = 0; j < c; ++j) { printf("Enter element a%d%d: ", i + 1, j + 1); scanf("%d", &a[i][j]); } printf("Enter elements of 2nd matrix:\n"); for (i = 0; i < r; ++i) for (j = 0; j < c; ++j) { printf("Enter element b%d%d: ", i + 1, j + 1); scanf("%d", &b[i][j]); } // adding two matrices for (i = 0; i < r; ++i) for (j = 0; j < c; ++j) { sum[i][j] = a[i][j] + b[i][j]; </pre>	5	CO1	L2

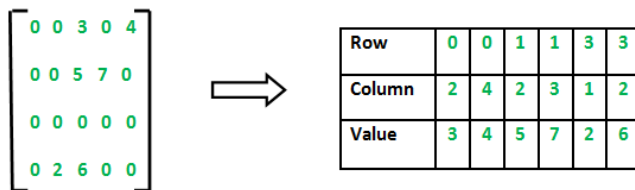
```

    }
    // printing the result
    printf("\nSum of two matrices: \n");
    for (i = 0; i < r; ++i)
        for (j = 0; j < c; ++j) {
            printf("%d  ", sum[i][j]);
            if (j == c - 1) {
                printf("\n\n");
            }
        }
    }
    return 0;
}

```

3 What is sparse matrix? Explain in detail different ways in which sparse matrix is represented.
 A matrix is a two-dimensional data object made of m rows and n columns, therefore having total m x n values. If most of the elements of the matrix have **0 value**, then it is called a sparse matrix.-2M
Method 1: Using Arrays:4M
 2D array is used to represent a sparse matrix in which there are three rows named as

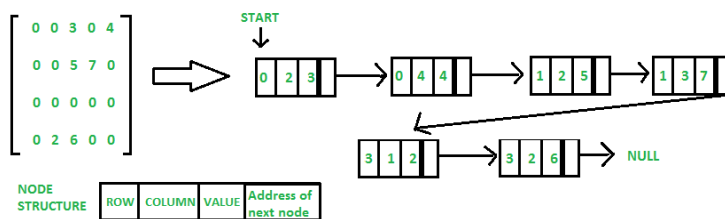
- **Row:** Index of row, where non-zero element is located
- **Column:** Index of column, where non-zero element is located
- **Value:** Value of the non zero element located at index – (row,column)



Method 2: Using Linked Lists-4M

In linked list, each node has four fields. These four fields are defined as:

- **Row:** Index of row, where non-zero element is located
- **Column:** Index of column, where non-zero element is located
- **Value:** Value of the non zero element located at index – (row,column)
- **Next node:** Address of the next node



4a Define an Array. Write C++ program to store elements and retrieve elements from an array.
 Array is a collection of elements of similar data type-1M
 PGR-4M

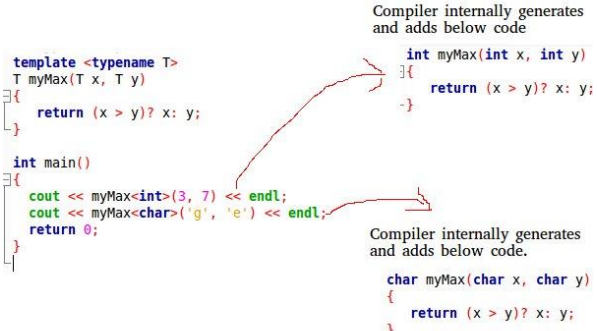
```

#include <iostream>

```

10	CO1	L2
5	CO1	L1

	<pre> using namespace std; int main() { int numbers[5]; cout << "Enter 5 numbers: " << endl; // store input from user to array for (int i = 0; i < 5; ++i) { cin >> numbers[i]; } cout << "The numbers are: "; // print array elements for (int n = 0; n < 5; ++n) { cout << numbers[n] << " "; } return 0; } </pre>			
<p>4b</p>	<p>Write C++ program to find factorial of a given number and Fibonacci series using function.</p> <p>FIBONACCI-3M</p> <pre> #include<stdio.h> int fib(int); void main() { int n,i; printf("Enter the nth term : \n"); scanf("%d",&n); printf("Fibonacci series is : "); for(i=0;i<n;i++) printf(" %d ",fib(i)); getch(); } int fib(int n) { if(n==0) return 0; if(n==1 n==2) return 1; else return fib(n-1)+fib(n-2); } </pre> <p>FACTORIAL-2M</p> <pre> #include<iostream> using namespace std; int main() { </pre>	<p>10</p>	<p>CO1</p>	<p>L3</p>

	<pre> int factorial(int); int fact,value; cout<<"Enter any number: "; cin>>value; fact=factorial(value); cout<<"Factorial of a number is: "<<fact<<endl; return 0; } int factorial(int n) { if(n<0) return(-1); /*Wrong value*/ if(n==0) return(1); /*Terminating condition*/ else { return(n*factorial(n-1)); } } </pre>			
5a	<p>Define template. Explain the syntax of template with suitable example.</p> <p>A template is a simple yet very powerful tool in C++. The simple idea is to pass the data type as a parameter so that we don't need to write the same code for different data types. For example, a software company may need to sort() for different data types. Rather than writing and maintaining multiple codes, we can write one sort() and pass the datatype as a parameter. 2M</p>  <pre> template <typename T> T myMax(T x, T y) { return (x > y)? x: y; } int main() { cout << myMax<int>(3, 7) << endl; cout << myMax<char>('g', 'e') << endl; return 0; } </pre> <p>Compiler internally generates and adds below code</p> <pre> int myMax(int x, int y) { return (x > y)? x: y; } char myMax(char x, char y) { return (x > y)? x: y; } </pre> <p>3M</p>	5	CO1	L2
5b	<p>Define function. Bring out the difference between local and global variables used in function.</p> <p>A function is a block of code that performs some operation. A function can optionally define input parameters that enable callers to pass arguments into the function. A function can optionally return a value as output.-1M</p> <p>The main difference between Global and local variables is that global variables can be accessed globally in the entire program, whereas local variables can be accessed only within the function or block in which they are defined.-2M</p> <p>Local variables are created when the function starts its execution and are lost when the function ends. Global variables, on the other hand, are created as execution of the program begins and are lost when the program is ended. In contrast to global variables, local variables do not offer data sharing.2M</p>	5	CO1	L2
6	<p>Write a C++ program with the help of template to input 2 numbers and output smallest number using class and object.10M</p>	10	CO1	L3

```
#include <iostream>
```

```
using namespace std;
```

```
#define SIZE 50 // Defining max size of array
```

```
int main()
```

```
{
```

```
int array[SIZE];
```

```
int i, max, min, size;
```

```
// Input size of the array
```

```
cout << "Enter size of the array: ";
```

```
cin >> size;
```

```
// Input array elements
```

```
cout << "\nEnter " << size << " elements in the array: ";
```

```
for (i = 0; i < size; i++)
```

```
cin >> array[i];
```

```
// Assume first element as maximum and minimum
```

```
max = array[0];
```

```
min = array[0];
```

```
// Find maximum and minimum in all array elements.
```

```
for (i = 1; i < size; i++)
```

```
{
```

```
// If current element is greater than max
```

```
if (array[i] > max)
```

```
max = array[i];
```

```
// If current element is smaller than min
```

```
if (array[i] < min)
```

```
min = array[i];
```

```
}
```

```
// Print maximum and minimum elements
```

```
cout << "\nMaximum element =" << max << "\n";
```

```
cout << "\nMinimum element =" << min;
```

```
return 0;
```

```
}
```


--	--	--	--	--

Course Instructor

CCI signature

HoD Signature