USN | | | | | | | | | | |

Internal Assessment Test 2 – June. 2022

Sub:	**Data Structure using C++**					Sub Code:	18EC643	Branch:	ECE
Date:	9-06-2022	Duration:	90 Minutes	Max Marks:	50	Sem / Sec:	**6 (A, B,C,D)**		**OBE**

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1	(a) Define sparse matrix and explain with an example how to represent sparse matrix using triplet concept.	(5+5)M	CO2	L2 L3

An $m \times n$ matrix is said to be **sparse** if "many" of its elements are zero. A mat
that is not sparse is **dense**. The boundary between a dense and a sparse mat
is not precisely defined. Diagonal and tridiagonal $n \times n$ matrices are sparse. E

Representation Using a Single Linear List

The nonzero entries of an irregular sparse matrix may be mapped into a
list in row-major order. For example, the nonzero entries of the 4 × 8 mat
Figure 7.14(a) in row-major order are 2, 1, 6, 7, 3, 9, 8, 4, 5.

To reconstruct the matrix structure, we need to record the originating rov
column for each nonzero entry. So each element of the array into which the s

0	0	0	2	0	0	1	0
0	6	0	0	7	0	0	3
0	0	0	9	0	8	0	0
0	4	5	0	0	0	0	0

(a) A 4 × 8 matrix

terms	0	1	2	3	4	5	6	7	8
row	1	1	2	2	2	3	3	4	4
col	4	7	2	5	8	4	6	2	3
value	2	1	6	7	3	9	8	4	5

(b) Its linear list representation

A sparse matrix and its linear list representation

(b) Write a C++ template based program to insert new element in given array or linearlist.

(c)

2	(a)What is linked list how it is different from array and also explain advantages of linked list? b) Write a C++ template for structure of ChainNode.	(5+5)M	CO2	L1 L3
3	(a) Explain with neat diagram of column mapping technique and also explain how to access element with mapping. (b) Write a C++ template for set function of diagonal matrix.	[6+4]M	CO2	L3 L3

```
template<class T>
void diagonalMatrix<T>::set(int i, int j, const T& newValue)
{// Store newValue as (i,j)th element.
   // validate i and j
   if (i < 1 || j < 1 || i > n || j > n)
       throw matrixIndexOutOfBounds();

   if (i == j)
     // save the diagonal value
      element[i-1] = newValue;
   else
      // nondiagonal value, newValue must be zero
      if (newValue != 0)
         throw illegalParameterValue
               ("nondiagonal elements must be zero");
}
```

Program 7.10 Set method for `diagonalMatrix`

4	**(a)** What is ADT in Data structure? Write Abstract data type specification of an array or linear list. **(b)**Write template based C++ program for get function of a tridiagonal matrix.	[4+6]M	CO2	L1 L3

7.3.3 Tridiagonal Matrix

In a `rows` × `rows` tridiagonal matrix, the nonzero elements lie on one of the three diagonals:

1. Main diagonal—for this, $i = j$.
2. Diagonal below main diagonal—for this, $i = j + 1$.
3. Diagonal above main diagonal—for this, $i = j - 1$.

The number of elements on these three diagonals is `3*rows-2`. We can use a one-dimensional array `element` with `3*rows-2` positions to represent the tridiagonal matrix. Only the elements on the three diagonals are explicitly stored. Consider the 4×4 tridiagonal matrix of Figure 7.8(b). There are 10 elements on the main diagonal and the diagonals just above and below the main diagonal. If these elements are mapped into `element` by rows, then `element[0:9]` = [2, 1, 3, 1, 3, 5, 2, 7, 9, 0]; if the mapping is by columns, `element` = [2, 3, 1, 1, 5, 3, 2, 9, 7, 0]; and if the mapping is by diagonals beginning with the lowest, then `element` = [3, 5, 9, 2, 1, 2, 0, 1, 3, 7]. As we can see, there are several reasonable choices for the mapping of T into `element`. Each requires a different code for the `get` and `set` methods. Suppose that the class `tridiagonalMatrix` maps by diagonals. The data members and constructor are quite similar to those of the class `diagonal`. Program 7.11 gives

```
template <class T>
T tridiagonalMatrix<T>::get(int i, int j) const
{// Return (i,j)th element of matrix.

   // validate i and j
   if ( i < 1 || j < 1 || i > n || j > n)
       throw matrixIndexOutOfBounds();

   // determine lement to return
   switch (i - j)
   {
      case 1: // lower diagonal
              return element[i - 2];
      case 0: // main diagonal
              return element[n + i - 2];
      case -1: // upper diagonal
              return element[2 * n + i - 2];
      default: return 0;
   }
}
```

Program 7.11 The method **get** for a tridiagonal matrix

5	(a) Write a C++ program to represent sparse matrix in triplet form.	[5+5]M	CO2	L3 L2

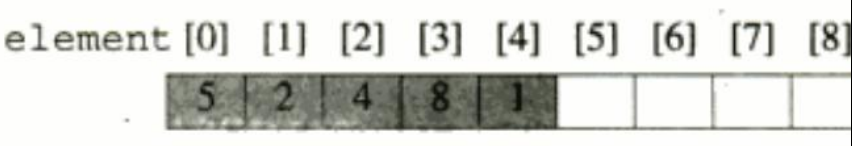
```
// C++ program for Sparse Matrix Representation
// using Array
#include <iostream>
using namespace std;

int main()
{
        // Assume 4x5 sparse matrix
        int sparseMatrix[4][5] =
        {
                {0 , 0 , 3 , 0 , 4 },
                {0 , 0 , 5 , 7 , 0 },
                {0 , 0 , 0 , 0 , 0 },
                {0 , 2 , 6 , 0 , 0 }
        };

        int size = 0;
```

```
        for (int i = 0; i < 4; i++)
                for (int j = 0; j < 5; j++)
                        if (sparseMatrix[i][j] != 0)
                                size++;

        // number of columns in compactMatrix (size) must
be
        // equal to number of non - zero elements in
        // sparseMatrix
        int compactMatrix[3][size];

        // Making of new matrix
        int k = 0;
        for (int i = 0; i < 4; i++)
                for (int j = 0; j < 5; j++)
                        if (sparseMatrix[i][j] != 0)
                        {
                                compactMatrix[0][k] = i;
                                compactMatrix[1][k] = j;
                                compactMatrix[2][k] =
sparseMatrix[i][j];
                                k++;
                        }

        for (int i=0; i<3; i++)
        {
                for (int j=0; j<size; j++)
                        cout <<" "<< compactMatrix[i][j];

                cout <<"\n";
        }
        return 0;
}
```

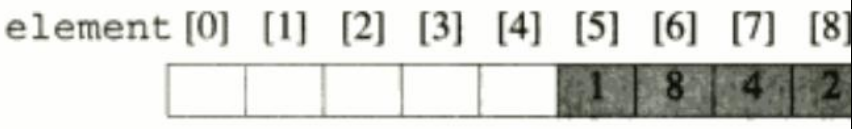
(b)What is linked list? Explain with neat diagram how to insert new element.

```
template<class T>
void arrayList<T>::insert(int theIndex, const T& t
{// Insert theElement so that its index is theInde
   if (theIndex < 0 || theIndex > listSize)
 if (listSize == arrayLength)
    {// no space, double capacity
       changeLength1D(element, arrayLength, 2 * a
       arrayLength *= 2;
    }

   // shift elements right one position
   copy_backward(element + theIndex, element + 1
                 element + listSize + 1);

   element[theIndex] = theElement;

   listSize++;
}
```

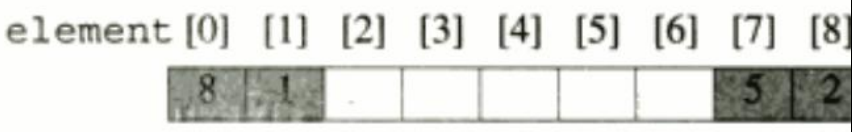
6	a) Describe different way of array representation with neat diagrams. b)	[6+4]M	CO2	L2 L3



(a) *location* (*i*)=*i*



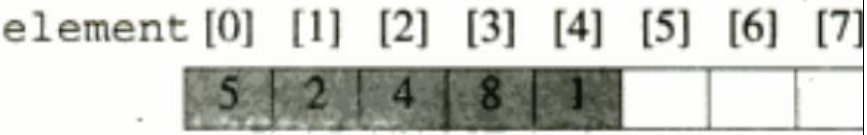
(b) *location* (*i*)=9 – *i*



(c) *location* (*i*)=(7 + *i*) % 10

$location(i) = \texttt{arrayLength} - i -$

stores the list elements backwards beginning at the right end of th
and the formula



(a) *location* (*i*)=*i*

	b)Write C++ template to change one dimension array.
	`template<class T>`
	`void changeLength1D(T*& a, int oldLength, int newLength)`
	`{`
	`if (newLength < 0)`
	`throw illegalParameterValue("new length must be >=`
	
	`T* temp = new T[newLength];  // new array`
	`int number = min(oldLength, newLength);  // number to`
	`copy(a, a + number, temp);`
	`delete [] a;  // deallocate`
	`a = temp;`
	`}`			

7	(a)Show a C++ template for linked list destructor.
	`template<class T>`
	`chain<T>::~chain()`
	`{// Chain destructor. Delete all nodes in chai`
	`while (firstNode != NULL)`
	`{// delete firstNode`
	`chainNode<T>* nextNode = firstNode->next`
	`delete firstNode;`
	`firstNode = nextNode;`
	`}`
	`}`
	(b)Write a C++ program to print sum of two sparse matrix (Consider input matrix as sparse matrix).
	.
		[4+6]	CO2	L3
				L3

HOD CI CCI