Karadeniz Technical University
Department of Computer Engineering
Lecturer Ömer ÇAKIR

COM 2005 Data Structures
Final Exam, 13.01.2025, 10:00
Duration : **100** Minutes

SOLUTIONS

```
void CircularlyLinkedList::
insertOrdered(CircularlyNode* newNode,
              CircularlyNode* back)
{
    if (newNode->score < back->next->score)
    {
        if (back->next == cursor)
        {
           newNode->next     = back->next->next;
           back->next->next  = newNode;
           cursor            = cursor->next;
           return;
        }

        insertOrdered(newNode, back->next);
    }
    else
    {
        newNode->next      = back->next;
        back->next         = newNode;
    }
}

int main()
{
   CircularlyLinkedList list;
   list.add("Rose", 590);

   CircularlyNode* newNode = new CircularlyNode;
   newNode->elem  = "Anna";
   newNode->score = 660;
   list.insertOrdered(newNode, list.cursor);

   newNode = new CircularlyNode;
   newNode->elem  = "Paul";
   newNode->score = 720;
   list.insertOrdered(newNode, list.cursor);

   newNode = new CircularlyNode;
   newNode->elem  = "Jill";
   newNode->score = 740;
   list.insertOrdered(newNode, list.cursor);

   newNode = new CircularlyNode;
   newNode->elem  = "Rob";
   newNode->score = 750;
   list.insertOrdered(newNode, list.cursor);

   newNode = new CircularlyNode;
   newNode->elem  = "Mike";
   newNode->score = 1105;
   list.insertOrdered(newNode, list.cursor);

   newNode = new CircularlyNode;
   newNode->elem  = "Jack";
   newNode->score = 510;
   list.insertOrdered(newNode, list.cursor);
}
```

1. How many times does the function **insertOrdered()** call itself recursively? **(30P)**

A)1 **B)3** ***C)5*** **D)7** **E)9**

```
void BinaryTree::eulerLike(TreeNode* v) const
{
	if (v->left != NULL)
	{
		cout << v->elem;
		eulerLike(v->left);
	}

	cout << v->elem;

	if (v->right != NULL)
	{
		eulerLike(v->right);
		cout << v->elem;
	}
}

int main()
{
	BinaryTree binaryTree;
	binaryTree.addNode(binaryTree.root,  8);
	binaryTree.addNode(binaryTree.root,  4);
	binaryTree.addNode(binaryTree.root, 12);
	binaryTree.addNode(binaryTree.root,  2);
	binaryTree.addNode(binaryTree.root,  6);
	binaryTree.addNode(binaryTree.root, 10);
	binaryTree.addNode(binaryTree.root, 14);

	binaryTree.eulerLike(binaryTree.root);
}
```

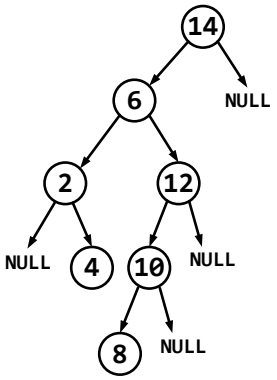
2. What is the output of the program above? **(20P)**

8	4	2	4	6	4	8	12	10	12	14	12	8

3. How many times does the function **eulerLike()** call itself recursively? **(20P)**

A)4 ***B)6*** **C)8** **D)10** **E)12**

(A) 8 4 12 2 6 10 14
(B) 8 4 12 2 10 6 14
(C) 8 4 12 6 2 10 14
(D) 8 4 12 10 2 6 14
(E) 8 4 12 6 10 2 14
(F) 8 4 12 10 6 2 14



4. Which two patterns produce the same **Splay** tree? **(30P)**

B	D