Karadeniz Technical University
Department of Computer Engineering
Lecturer Ömer ÇAKIR

COM 2005 Data Structures
Midterm Exam, 17.11.2016, 15:00, D-1, D-8
Duration : **90** Minutes

		EXAM GRADE	
NUMBER :	**NAME :**		
Rules to be Obeyed During the Exam	**SIGNATURE :**	[........]	....

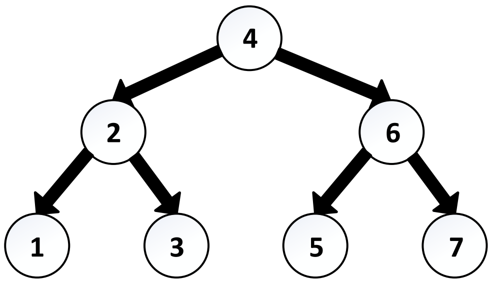
1. Cell phones are not allowed to be used as a calculator or a watch. They must be switched off and placed in the pocket.
2. Brief information about the exam will be given at the begining, then no one is not allowed to ask a question during the exam.
3. Do not to forget to sign this paper after writing your number and name.

```
void traverse(Node* v)
{
       cout << v->elt << " ";

       if (v->left != NULL)
       {                                    // A
              traverse(v->left);
              cout << v->elt << " ";
       }                                    // A

       if (v->right != NULL)
       {                                    // B
              traverse(v->right);
              cout << v->elt << " ";
       }                                    // B
}
```

1. Supposing that the **traverse()** function is called in the **main()** function with the **root** of the tree below:

a) What is the output of the **traverse()** function if the lines (curly phrases) indicated by **// A** are commented out? **(15P)**

b) What is the output of the **traverse()** function if the lines (curly phrases) indicated by **// B** are commented out? **(15P)**



```
void addBack(const string& e, const int& i)
{
   CircularlyNode* v = new CircularlyNode;
   v->elem = e;
   v->score = i;

   if (cursor == NULL)
   {
      v->next  = v;
      cursor   = v;
   }
   else
   {
      .................... = .................... ;

      .................... = .................... ;

      .................... = .................... ;

   }
}
```

2. Complete the **addBack()** function that adds a node at the end of a circularly linked list. **(20P)**

```cpp
void insertOrdered(const string& e, const int& i)
{
  DoublyNode* newNode = new DoublyNode;
  newNode->elem       = e;
  newNode->score      = i;

  DoublyNode* current = header;

  while (current->next != trailer)
  {
     if(newNode->score >= current->next->score)
        current = current->next;
     else
        break;
  }


  newNode->next          = ......................;

  newNode->prev          = ......................;

  ...................... = newNode;

  ...................... = newNode;

}

int main()
{
  DoublyLinkedList list;

  list.insertOrdered("Paul",  720);
  list.insertOrdered("Rose",  590);
  list.insertOrdered("Anna",  660);
  list.insertOrdered("Mike", 1105);
  list.insertOrdered("Rob" ,  750);
  list.insertOrdered("Jack",  510);
  list.insertOrdered("Jill",  740);

}
```

3. Complete the **insertOrdered()** function. **(30P)**

```cpp
#include <iostream>
#include <stack>
using namespace std;

void main()
{
	stack<int> stl_stack;

	int temp = 61;

	while (temp != 0)
	{
		stl_stack.push(temp % 2);
		temp = temp / 2;
	}

	while (!stl_stack.empty())
	{
		if (stl_stack.top() == 1)
			cout << '1';
		else
			cout << '0';

		stl_stack.pop();
	}

	getchar();
}
```

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