Karadeniz Technical University
Department of Computer Engineering
Lecturer Omer CAKIR

COM 205 Data Structures
Midterm Exam, 19.11.2013, 13:00, D-1, D-8
Time : **90** Minutes

NUMBER : ..... **NAME :**	**EXAM GRADE**	
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 print1(DoublyNode* node)
{
    cout << node->elem << node->score << endl;

    if (node->next == trailer) return;

    print1(node->next);
}

void print2(DoublyNode* node)
{
    if (node == trailer) return;

    cout << node->elem << node->score << endl;

    print2(node->next);
}

void main()
{
    DoublyLinkedList list;

    list.insertOrdered("Paul", 720); //küçükten
    list.insertOrdered("Rose", 590); //büyüğe
    list.insertOrdered("Anna", 660); //sıralı ekle

    list.print1(list.header->next);
    list.print2(list.header->next);
}
```

1.

a) Write down the output of the **print1()** and the **print2()** function calls above? **(20P)**

print1()	print2()

b) How many times do the **print1()** and **print2()** functions call themselves recursively? **(10P)**

print1() calls itself recursively ..... times.

print2() calls itself recursively ..... times.

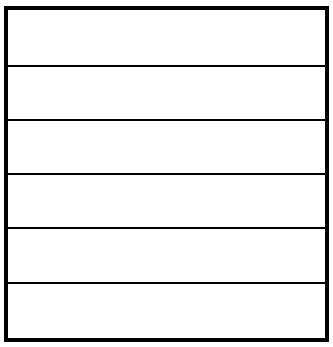
```
void quadruple(int A[], int i, int n)
{
    if (n == 1) cout << A[i] << endl;
    else
    {
       quadruple( A, i +   n/4, n/4 );
       quadruple( A, i        , n/4 );
       quadruple( A, i + 3*n/4, n/4 );
       quadruple( A, i + 2*n/4, n/4 );
    }
}

void main()
{
  int A[16]={1,2,3,4,5,6,7,8,
     9,10,11,12,13,14,15,16};

  quadruple(A, 0, 16);
}
```

Output

2. Write up-right side the output of the program above? **(30P)**

```
SinglyNode* SinglyLinkedList::funcA(SinglyNode* node)
{
   if(node->next == NULL) return node;
   else funcA(node->next);
}

void SinglyLinkedList::funcB(SinglyNode* node)
{
   if(node->next == NULL)
   {
      delete head;
      head = NULL;
      return;
   }

   if (node->next->next == NULL)
   {
      delete node->next;
      node->next = NULL;
      return;
   }

   funcB(node->next);
}

SinglyLinkedList* SinglyLinkedList::funcC()
{
  SinglyLinkedList* newList = new SinglyLinkedList();

  SinglyNode* node           = funcA(head);
  newList->head              = new SinglyNode();
  newList->head->elem        = node->elem;
  newList->head->score       = node->score;
  SinglyNode* tempHead       = newList->head;
  funcB(head);

  while(head != NULL)
  {
     node                    = funcA(head);
     tempHead->next          = new SinglyNode();
     tempHead->next->elem    = node->elem;
     tempHead->next->score   = node->score;
     tempHead                = tempHead->next;
     funcB(head);
  }
  tempHead->next     = NULL;

  return newList;
}

void main()
{
  SinglyLinkedList* list = new SinglyLinkedList();

  list->insertOrdered("Mike", 1105);
  list->insertOrdered("Rob",  750);
  list->insertOrdered("Paul", 720);
  list->insertOrdered("Anna", 660);
  list->insertOrdered("Rose", 590);
  list->insertOrdered("Jack", 510);

  SinglyLinkedList* newList = list->funcC();
  newList->print();

  ::getchar();
}
```

3. Write down the output of the program on the left side? **(20P)**



What does funcA do? (1 sentence) (5P)

What does funcB do? (1 sentence) (5P)

What does funcC do? (1 sentence) (10P)