1. What does fList() do? Explain your answer. (30P)

```cpp
DoublyLinkedList* fList(DoublyLinkedList* list1)
{
    DoublyLinkedList* list2 = new DoublyLinkedList();
    DoublyNode* nodeA = NULL;
    DoublyNode* nodeB = NULL;

    while(!list1->empty())
    {
        nodeA = list1->header->next;
        nodeB = list1->header->next->next;

        while (nodeB != list1->trailer)
        {
            if (nodeB->score < nodeA->score)
            {
                nodeA = nodeB;
                nodeB = nodeB->next;
            }
            else
            {
                nodeB = nodeB->next;
            }
        }

        list2->addBack(nodeA->elem, nodeA->score);
        list1->remove(nodeA);
    }

    return list2;
}
```

2. What is the output of the program above? (30P)

```cpp
int bic(int n, int k)
{
    if (k == 0) return 1;
    if (k == n) return 1;
    return bic(n - 1, k - 1) + bic(n - 1, k);
}
```

```cpp
void main()
{
    for (int n = 0; n < 5; n++)
    {
        for (int k = 0; k <= n; k++)
        {
            cout << bic(n, k) << " ";
        }
        cout << endl;
    }
}
```
```cpp
bool empty()
{
    return (header->next == trailer);
}

void addFront(const string& e, const int& i)
{
    add(header->next, e, i);
}

void add(DoublyNode* v, string& e, int& i)
{
    DoublyNode* u = new DoublyNode;
    u->elem = e;
    u->score = i;
    ....
    
    DoublyNode* first = header;
    while (!first->next == trailer)
    {
        cout << first->next->elem << \
             "t" << first->next->score << endl;
        first = first->next;
    }
}

void printH2T()
{
    if (empty())
    {
        cout << "List is empty !" << endl;
        return;
    }

    DoublyNode* first = header;
    while (!first->next == trailer)
    {
        cout << first->next->elem << \
             "t" << first->next->score << endl;
        first = first->next;
    }
}

void main()
{
    DoublyLinkedList list;
    list.addFront("Rob", 750);
    list.addFront("Paul", 720);
    list.printH2T();
}
```

3. Taking into account the lines represented by ..... in the function add() answer the following choices:

i) (10P) (You’ll lose 5Ps from each wrong answer)
If the lines are like these

```cpp
u->next = v;
u->prev = v->prev;
v->prev = u; 
v->prev->next = u;
```

the printH2T() function:
(A) will print the list elements.
(B) will print "List is empty !".
(C) will enter into an infinite loop.

ii) (10P) (You’ll lose 5Ps from each wrong answer)
If the lines are like these

```cpp
u->next = v;
v->prev->next = u;
u->prev = v->prev;
v->prev = u;
```

the printH2T() function:
(A) will print the list elements.
(B) will print "List is empty !".
(C) will enter into an infinite loop.

iii) (10P) (You’ll lose 5Ps from each wrong answer)
If the lines are like these

```cpp
u->next = v;
v->prev->next = u;
v->prev = v->prev;
```

the printH2T() function:
(A) will print the list elements.
(B) will print "List is empty !".
(C) will enter into an infinite loop.

iv) (10P) (You’ll lose 5Ps from each wrong answer)
If the lines are like these

```cpp
u->next = v;
v->prev = u; 
u->prev = v->prev;
```

the printH2T() function:
(A) will print the list elements.
(B) will print "List is empty !".
(C) will enter into an infinite loop.