



```

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

    if (cursor == NULL)
    {
        newNode->next = newNode;
        cursor = newNode;
        return;
    }

    if( i < cursor->next->score)
    {
        .....;
        .....;
        return;
    }

    if( i > cursor->score)
    {
        .....;
        .....;
        .....;
        return;
    }

    CircularlyNode* front = cursor->next;
    CircularlyNode* back = NULL;

    while( newNode->score > front->score )
    {
        back = front;
        front = front->next;
    }

    .....;
    .....;
}

```

2. Düğümleri dairesel bağlı listeye **score** değerlerine göre küçükten büyüğe sıralı ekleyen **insertOrdered()** fonksiyonunda ..... ile temsil edilen satırlara gereken kodları ekleyiniz. (35P)

```

void gList(DoublyLinkedList* list,
           DoublyNode* hNext, DoublyNode* tPrev)
{
    if (hNext == tPrev) return;

    if (hNext->next == tPrev)
    {
        list->add(hNext, tPrev->elem, tPrev->score);
        list->remove(tPrev);
        return;
    }
    else
    {
        list->add(tPrev, hNext->elem, hNext->score);
        hNext = hNext->next;
        list->remove(hNext->prev);

        list->add(hNext, tPrev->elem, tPrev->score);
        tPrev = tPrev->prev;
        list->remove(tPrev->next);

        gList(list, hNext, tPrev->prev);
    }
}

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

    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);

    gList(list,
          list->header->next,
          list->trailer->prev);
}

```

3. a) **gList()** fonksiyonu ne iş yapar? Kısaca açıklayınız. (15P)  
**İpucu** → **add()** fonksiyonu 2. ve 3. parametre ile üretilen düğümü 1. parametreden önce (prev) ekler.


b) **if(hNext->next == tPrev){...}** bloğunun alternatifi nedir? (20P)

```

if ( hNext->next == tPrev )
{
    .....;
    .....;
    return;
}

```