



SOLUTIONS

```
void insertOrdered(
    SinglyNode* newNode,
    SinglyNode* previous, SinglyNode* current)
{
    if ((current == NULL)
        || (newNode->score <= current->score))
    {
        newNode->next = current;
        previous->next = newNode;
    }
    else
        insertOrdered(newNode, current,
                      current->next);
}

int main()
{
    SinglyLinkedList list;
    SinglyNode* newNode;

    list.head = new SinglyNode;
    list.head->elem = "NoName";
    list.head->score = 0;
    list.head->next = NULL;

    newNode = new SinglyNode;
    newNode->elem = "...";
    newNode->score = ....;
    list.insertOrdered(newNode, list.head, list.head);

    newNode = new SinglyNode;
    newNode->elem = "...";
    newNode->score = ....;
    list.insertOrdered(newNode, list.head, list.head);

    newNode = new SinglyNode;
    newNode->elem = "...";
    newNode->score = ....;
    list.insertOrdered(newNode, list.head, list.head);

    newNode = new SinglyNode;
    newNode->elem = "...";
    newNode->score = ....;
    list.insertOrdered(newNode, list.head, list.head);
}
```

1. Which score order inserts nodes with **min** recursive call?
(25P) *You'll loose 5P from wrong answer.*

- (A) 590, 660, 720, 1105
- (B) 660, 590, 720, 1105
- (C) 590, 1105, 660, 720
- (D) 1105, 720, 590, 660
- (E) 1105, 720, 660, 590

2. What is the number of recursive **insertOrdered()** calls of each choice in question 1? (25P)

- | | | | | | | | | | |
|---|----|---|---|---|---|---|---|---|---|
| A | 10 | B | 9 | C | 8 | D | 5 | E | 4 |
|---|----|---|---|---|---|---|---|---|---|

```
int main()
{
    CircularlyLinkedList Queue;

    Queue.enqueue(5);
    Queue.enqueue(4);
    Queue.enqueue(3);
    Queue.enqueue(2);
    Queue.enqueue(1);

    Queue.dequeue();
    Queue.dequeue();
    Queue.dequeue();

    Queue.enqueue(3);
    Queue.enqueue(4);
    Queue.enqueue(5);

    Queue.C.print();
}
```

3. What is the output of the program above? (25P)



```
int main()
{
    LinkedStack LStack;

    LStack.push(5);
    LStack.push(4);
    LStack.push(3);
    LStack.push(2);
    LStack.push(1);

    cout << "Top = " << LStack.top() << endl;

    LStack.pop();
    LStack.pop();
    LStack.pop();

    LStack.push(1);
    LStack.push(2);
    LStack.push(3);

    cout << "Top = " << LStack.top() << endl;
}
```

4. What is the output of the program above? (25P)

