1 [20 points]

Trace the following program and write the output.

```java
int n = 2;
while (n < 6)
{
    for (int k = n; k >= 0; k --)
    {
        if (k % 2 == 0)
            System.out.print("@");
        else
            System.out.print("*");
    }
    System.out.println();
    n ++;
}
```

2 [20 points]

Answer the following questions:

1. How do you measure the performance of a program?

2. The following program tries to find the maximum value and its index in an array \( a \). Justify its correctness using loop invariants.

```java
int max = a[0];
int id = 0;
for (int i = 0; i < a.length; i++)
{
```
if (a[i] >= max)
{
    max = a[i];
    id = i;
}

3  [20 points]

Write a recursive function to reverse the order of elements in a 1D int array. For example, if the input array has elements 1 2 3 4, your function will reverse the order of the array so that the array becomes 4 3 2 1.

4  [20 points]

Write a method to move the head node in a linked list to the end of the list. The linked list class has been partially implemented as follows:

```java
public class List
{
    class Node
    {
        int data;
        Node next;
        public Node(int data)
        {
            this.data = data;
            next = null;
        }
    }

    Node head;
    public List() { head = null; }
}
```

Your method has the signature:

```java
public int moveHeadToEnd()
```

You DO NOT have to write the class. Just complete the method so that it fits into the class List. You can use the following code to traverse the list and find the last node in the list.

```java
Node t = head;
while (t != null)
{
    t = t.next;
}
```
Consider the following code segment, determine the output of the program to screen when the program is running.

```java
public class Base {
    public int data;

    public Base() {
        data = 10;
    }

    public void doThis() {
        data++;
    }

    public void doThat() {
        data--;
    }

    public int getData() {
        return data;
    }
}

public class DerivedA extends Base {
    public DerivedA() {
        data = 1;
    }

    public void doThis() {
        data = data * 10;
    }

    public void doThat() {
        data = data / 10;
    }
}
```
public class DerivedB extends Base
{
    public DerivedB()
    {
        data = 30;
    }

    public void doThat()
    {
        data = data + 5;
    }
}

public class InheritancePlay
{
    public static void main(String[] args)
    {
        final int NUM_OBJECTS = 3;
        int index;
        Base[] hierarchicalObjects = new Base[NUM_OBJECTS];
        hierarchicalObjects[0] = new Base();
        hierarchicalObjects[1] = new DerivedA();
        hierarchicalObjects[2] = new DerivedB();
        for( index = 0; index < NUM_OBJECTS; index++ )
        {
            hierarchicalObjects[index].doThis();
            hierarchicalObjects[index].doThat();
        }

        for( index=0; index < NUM_OBJECTS; index++ )
        {
            System.out.println( hierarchicalObjects[index].getData() );
        }
    }
}

—— YOU MAY STOP HERE OR CONTINUE TO GET THE BONUS POINTS ——

6  Bonus Question[20 points]

Write a recursive program to merge 2 sorted integer arrays. Each array has a sequence of numbers in the ascending order. After merging the two arrays, you should generate an array that contains all the numbers from the two arrays and is in an ascending order. For example, one array contains {1, 3, 4, 9, 10} and the other {2, 4, 7, 11, 98, 100}. The merging result should be {1, 2, 3, 4, 4, 7, 9, 10, 11, 98, 100}. 