Peer Instruction

Recursion

What does the recursive code below print when called with d = 100.0 and n = 6?

```
public static double compute(double d, int n) {
  // base case
  if (n == 0) return 0.0;
  // compute term
  double term = d;
  // recursion call
  return term + compute(d / 10.0, n - 1);
}
```

A. 0.0
B. 100.0
C. 101.01
D. 111.11
E. 111.111

Recursion Examples

What does the recursive code below print when called with d = 100.0 and n = 6?

```
public static double compute(double d, int n) {
  // base case
  if (n == 0) return 0.0;
  // compute term
  double term = d;
  // recursion call
  return term + compute(d / 10.0, n - 1);
}
```

A. 0.0
B. 100.0
C. 101.01
D. 111.11
E. 111.111

Recursion Examples

What does the recursive code below print when called with s = "aabbccddeeff"?

```
public static String munge(String s) {
  if (s == null || s.length() <= 1)  // base case
    return s;
  else if (s.charAt(0) == s.charAt(1))
    return munge(s.substring(1, s.length()));
  else
    return s.charAt(0) +
        munge(s.substring(1, s.length()));
}
```

A. aabbccddeeff
B. abcdefabcdef
C. ababcdcdefef
D. abcdef
E. empty string

Recursion Examples

What does the recursive code below print when called with s = "aabbccddeeff"?

```
public static String munge(String s) {
    if (s == null || s.length() <= 1)  // base case
        return s;
    else if (s.charAt(0) == s.charAt(1))
        return munge(s.substring(1, s.length()));
    else
        return s.charAt(0) +
                munge(s.substring(1, s.length()));
}
```

A. aabbccddeeff
B. abcdefabcdef
C. ababcdcdefef
D. abcdef
E. empty string

Recursive Examples

What does the recursive code below print when called with number = 13 and base = 2?

```
public static void convert(int number, int base) {
    int remainder = number % base;
    int quotient = number / base;
    if (quotient > 0) convert(quotient, base);
    System.out.print(remainder);
}
```

A. 1111011
B. 11011
C. 1101
D. 1011
E. 110

Recursive Examples

What does the recursive code below print when called with number = 13 and base = 2?

```
public static void convert(int number, int base) {
    int remainder = number % base;
    int quotient = number / base;
    if (quotient > 0) convert(quotient, base);
    System.out.print(remainder);
}
```

A. 1111011
B. 11011
C. 1101
D. 1011
E. 110

Recursive Examples

On to the lecture

What does the following code print when i = 1?

```
public static void rec1 (int i) {
    if (i==0)
        System.out.print(i + " ");
    for (int j=0; j<2; j++) {
        rec1(i-1);
        rec1(i-1);
    }
}
```

A. 1 0
B. 0 0
C. 0 1
D. 0 0 0 0
E. None of the above

What does the following code print when i = 1?

```
public static void rec1 (int i) {
    if (i==0)
        System.out.print(i + " ");
    for (int j=0; j<2; j++) {
        rec1(i-1);
        rec1(i-1);
    }
}
```

A. 1 0
B. 0 0
C. 0 1
D. 0 0 0 0
E. None of the above

Stack overflow

What does the following code print when i = 1?

```
public static void rec2 (int i) {
    if (i==0) {
        System.out.print(i + " ");
    } else {
        for (int j=0; j<2; j++) {
            rec2 (i-1);
            rec2 (i-1);
        }
    }
}
```

A. 1 0
B. 0 0
C. 0 1
D. 0 0 0 0
E. None of the above

What does the following code print when i = 1?

```
public static void rec2 (int i) {
    if (i==0) {
        System.out.print(i + " ");
    } else {
        for (int j=0; j<2; j++) {
            rec2 (i-1);
            rec2 (i-1);
        }
    }
}
```

A. 1 0
B. 0 0
C. 0 1
D. 0 0 0 0
E. None of the above

What does the following code print when list = {1,3,5,7,9}

```
public int rec(int [] list){
        return rec3(list, 0);
}
public int rec3(int [] list, int start){
        if (start == list.length - 1) {
                return list[start];
        } else {
                return Math.max(list[start],
                        rec3(list, start + 1));
        }
}
```

A. 1
B. 3
C. 5
D. 7
E. 9

What does the following code print when list = {1,3,5,7,9}

```
public int rec(int [] list){
        return rec3(list, 0);
}
public int rec3(int [] list, int start){
        if (start == list.length - 1) {
                return list[start];
        } else {
                return Math.max(list[start],
                        rec3(list, start + 1));
        }
}
```

A. 1
B. 3
C. 5
D. 7
E. 9