

loop multiple choice worksheet #1

1. The following code segment intends that a user will enter a list of positive integers at the keyboard and terminate the list with a sentinel value:

```
int value;
final int SENTINEL = -999;
while (value != SENTINEL)
{
    // code to process the inputted value
    value = <integer value inputted by user>
}
```

The code segment is not correct. Which is a true statement?

- A. The sentinel gets processed.
- B. The last non-sentinel value entered in the list fails to get processed.
- C. A poor choice of SENTINEL causes the loop to terminate before all values have been processed.
- D. Running the program with this code causes a compile error.
- E. Entering the SENTINEL value as the first value causes a run-time error.

2. Which best describes method mystery?

```
public static int mystery(int x, int y)
// precondition: x > y
{
    int i = 1;
    int m = x;
    while (m % y != 0)
    {
        i++;
        m = i * x;
    }
    return m;
}
```

- A. It returns the smallest common factor of x and y , that is, the smallest positive integer divisor of both x and y .
- B. It returns the greatest common factor of x and y , that is, the largest integer divisor of both x and y .
- C. It returns the least common multiple of x and y , that is, the smallest integer that has both x and y as a factor.
- D. It returns y raised to the x^{th} power, that is y^x .
- E. It returns x raised to the y^{th} power, that is x^y .

3. Consider the following code segment:

```
int newNum = 0;
int temp = 0;
int num = <some integer value >= 0>
while (num > 10)
{
    temp = num % 10;
    num /= 10;
    newNum = newNum * 10 + temp;
}
System.out.println(newNum);
```

Which is a true statement about the segment?

- I. If $100 \leq \text{num} \leq 1000$ initially, the final value of `newNum` must be in the range $10 \leq \text{newNum} \leq 100$.
- II. There is no initial value of `num` that will cause an infinite `while` loop.

III. If num \leq 10 initially, newNum will have a final value of 0.

- A. I only
- B. II only
- C. III only
- D. II and III only
- E. I, II, and III

4. Consider the method reverse:

```
// precondition: n > 0
// postcondition: returns n with its digits reversed
// Example: If n = 234, method reverse returns 432
public static int reverse(int n)
{
    int rem = 0;
    int revNum = 0;

    <code segment>

    return revNum;
}
```

Which of the following replacements for *<code segment>* would cause the method to work as intended?

I.

```
for (int i = 0; i <= n; i++)
{
    rem = n % 10;
    revNum = revNum * 10 + rem;
    n /= 10;
}
```

II.

```
while (n != 0)
{
    rem = n % 10;
    revNum = revNum * 10 + rem;
    n /= 10;
}
```

III.

```
for (int i = n; i != 0; i /= 10)
{
    rem = i % 10;
    revNum = revNum * 10 + rem;
}
```

- A. I only
- B. II only
- C. I and II only
- D. II and III only
- E. I and III only