**Directions:** Determine the answer to each of the following questions or incomplete statements, using the available space for any necessary scratch work. Then decide which is the best of the choices given and fill in the corresponding oval on the answer sheet.

1. What is a *recursive method* ?  
     
   (A) A method that describes actions to be repeated using a loop.

(B) A method that describes actions to be repeated using a method that calls itself.

(C) A method that executes forever (infinitely).

(D) A method that uses binary search.

(E) A method that may or may not include a base case.

1. What is an *iterative method* ?  
     
   (A) A method that describes actions to be repeated using a loop.

(B) A method that describes actions to be repeated using a method that calls itself.

(C) A method that executes forever (infinitely).

(D) A method that uses binary search.

(E) A method that may or may not include a base case.

1. What is a *base case* ?  
     
   (A) A case within a recursive solution that involves reducing the overall problem to a simpler problem of the same kind that can be solved by a recursive call.

(B) An instance method that provides information about the state of an object, without  
 modifying it.

(C) A case within a recursive solution that is so simple, it can be solved without needing to call the method again.

(D) The simplest case that cannot be factored any further.

(E) The code that you build the rest of your method upon.

1. What is the *recursive case* ?  
     
   (A) A case within a recursive solution that involves reducing the overall problem to a simpler problem of the same kind that can be solved by a recursive call.

(B) An instance method that provides information about the state of an object, without  
 modifying it.

(C) A case within a recursive solution that is so simple, it can be solved without needing to call the method again.

(D) The simplest case that cannot be factored any further.

(E) The code that you build the rest of your method upon.

1. What will cause *infinite recursion*?

(A) Leaving out the recursive case in your recursive solution.

(B) Leaving out the recursive case in your iterative solution.

(C) Leaving out the base case in your iterative solution.

(D) Leaving out the recursive case in your recursive solution.

(E) Leaving out the iterative case in your recursive solution.

Questions 6 – 9 refer to the following code:

public int fibonacci(int n){

if(n == 0)

return 0;

else if(n == 1)

return 1;

else

return fibonacci(n - 1) + fibonacci(n - 2);

}

6. The code above is an example of:

(A) Code that executes infinite recursion.

(B) Code that loops infinitely.

(C) Recursive code.

(D) Iterative code.

(E) A call stack.

7. The call fibonacci(2) causes the Fibonacci method to output:

(A) 0

(B) 1

(C) 2

(D) 3

(E) “fibonacci(1) + fibonacci(0)”

8. The call fibonacci(8) causes the fibonacci method to output:

(A) 0

(B) 5

(C) 8

(D) 21

(E) “fibonacci(1) + fibonacci(0)”

9. What potential problems may arise from using the fibonacci method as shown?

(A) The program won’t compile.

(B) It is infinitely recursive on all inputs.

(C) The code will take longer and longer to execute as the numbers in the sequence increase.

(D) There is a code path that does not return a value.

(E) There is no base case in the version shown.

10. In the space below, write recursive code to compute the sum of integers from 0 to N.