```
public class Calculator{

    private int memory1, memory2;
    public double pi = 3.1415;

    public Calculator(){
        System.out.println("Calculator created!");
        memory1 = 0; memory2 = 0;
    }

    public String toString() {
        return("Calculator Version 1.0");
    }

    public int add(int a, int b){
        int answer = a + b;
        return(answer);
    }

    public int add(){
        int answer = memory1 + memory2;
        return(answer);
    }

    public void setMemory1(int a){
        memory1 = a;
    }

    public void setMemory2(int b){
        memory2 = b;
    }

    public void showMemory(){
        System.out.println("Memory1 is: " + memory1);
        System.out.println("Memory2 is: " + memory2);
    }

    public int getMemory1() {
        return(memory1);
    }

    public int getMemory2() {
        return(memory2);
    }


} //end of class
```

```
public class Card{

    private cardNum;
    private int value;
    private String suit;

    public Card(){
        System.out.println("Default Card created");
        cardNum = 0; value = 0; suit = "NA";
    }

    public Card(int num){
        System.out.println("Card Created");
        cardNum = num;
        setValue();
        setSuit();
    }

    public void setValue(){
        if ((cardNum <= 0) || (cardNum>=53) )
            value = 0;
        else{
            value = (cardNum-1) % 13 + 1;
            if (value > 10)
                value = 10;
            }
    }

    public int getValue(){
        return(value);
    }

    public void setSuit(){
        if ((cardNum <= 0) || (cardNum >=53))
            suit = "NA";
        else{
            int suitnum = (cardNum-1) / 13;
            if (suitnum == 0)
                suit = "CLUBS";
            else if(suitnum == 1)
                suit = "DIAMONDS";
            else if(suitnum == 2)
                suit = "SPADES";
            else
                suit = "HEARTS";
        }
    }

    public String getSuit(){
        return(suit);
    }

}//end of class
```

Quick Questions Calculator Class

1. Which method is the constructor for the Calculator class?

   public Calculator()

   Compare the constructor method to the other methods.
   There is something different. What is this difference?

   no return type (no void, int, double, String, etc) in front of the method name
   also exact same name as class

2. How many methods in the Calculator class require parameters to operate?

   3

3. In a program, how would you instantiate an instance of the Calculator class with the name *C* ?
   (in other words, how do you create a Calculator called *C* ?)

   ```
   Calculator C = new Calculator();
   //another way
   Calculator C;
   C = new Calculator();
   ```

4. Assume that you have created a Calculator object called *C*. Give an example of how you would set memory slot 2 to store the value 77.

   C.setMemory2(77);

5. Using *C*, give an example of printing out one of its variables that does not violate public/private access rules.

   System.out.println(C.pi) ;

6. Using *C*, give an example of printing out one of its variables that violates public/private access rules.

   System.out.println(C.memory1) ;

7. What does the *void*, *int, double*, *String* in front of the method name tell you?

   What type of value the method will return.
   Void means no value is going to be returned (it just does code).

8. How many methods in the Calculator class return values?

   4

9. How could you use *C* to add the numbers 10 and 20 together and print out the answer?

   ```
   int sum = C.add(10,20) ;
   //another way
   System.out.println( C.add(10,20) ) ;
   ```

10. The Calculator class has two methods called *add* (overloaded methods). How does the program know which add method to run when a programmer uses one of the add methods?

It looks at what parameters (arguments) you are passing in. If you pass no arguments, it uses the add method with no arguments. If you pass in two integer arguments it uses the method with two integer arguments.

11. What does the *void* keyword tell you about a method? **See answer 7.**

12. What affect does declaring a variable as *private* have on the variable's usage?

    Cannot access them from outside the class they were made in (Calculator objects could use them but no other class, especially the runner classes cannot!).

13. Assume the methods *setMemory1* did not exist. Would you ever be able to change the value of the member variable *memory1* ? If so, how?

    Nope. No way to change it in a runner class since it is private.

14. What would be the output of the following: *System.out.println(C);*

    This would invoke the *toString* method of the class and print out "Calculator Version 1.0"

15. Will the line:

    C.setMemory1(1); C.setMemory2(2);
    System.out.println(C.add(5,15) + C.add(C.add(5,3) , 8));

    compile and run and output anything? If so, what?

    Yes, it will work fine. A method that returns an integer can be treated just like any other integer value. I'm too old to do the math, you add those numbers up!

16. Do you see any problems that might occur with the line below? If so, what?

    int num = C.add(1.35, 28.42);

    The add method that takes in two parameters requires integers. Since you are sending it doubles the compiler will probably give an error telling you that you will lose precision on the values. You would have to do something like this: C.add((int)1.35, (int)28.42) ;

17. ***A programmer wants to give the calculator class the ability to do 'powers' with integers, like
    5 to the power of 2 returns 25, 2 to the power of 5 returns 32, 4 to the power of 3 returns 64, etc.
    Write a method called *power* that would accept TWO integer parameters and return the correct integer answer.

```
public int power(int a, int b)  {          //going to solve a to the power of b
 if (b==0)                                 //notice you can have several return statements!
 return(1);                          //once you return, you are gone, out of the method.
 int c=a;
 for(int k=2; k<=b; k++)
 c=c*a;

 return(c);
}
```

18. Give an example of a mutator method and an accessor method found in the Calculator class.
mutator: setMemory1 , accessor: getMemory1

Quick Questions Card Class

1. How many different constructors are there for the Card class?

   2

2. Give an example of how you could would create an instance of the Card class using each constructor.

   Card C = new Card()
   Card C2 = new Card(25)

3. Assume that you have created a Card called *card* in a program with the line
   Card card = new Card(20);

   What will the three member variables of the class be equal to after the constructor finishes executing it's code?

   cardNum: 20
   value: 7 *(follow the code inside of setValue)*
   suit: diamonds

4. Give an example of a line of code in a program that would violate public/private access rules for *card*.

   Using card.value, card.suit, or card.cardNum

5. How would you print out the suit of *card*?

   System.out.println(card.getSuit()) ;
   //or
   String s = card.getSuit() ;
   System.out.println(s) ;

6. How would you print out the cardNum of *card*?

   With magic. There is no way to do this since it is private and there have been no accessor methods provided.

7. Two cards are created the following way:
   Card c1 = new Card(Random.nextInt(52) + 1);
   Card c2 = new Card(Random.nextInt(52) + 1);

   A flush occurs if both the cards are the same suit. How would you check this?

   if ((c1.getSuit().equals(c2.getSuit() == true)
   //or
   String s1 = c1.getSuit();
   String s2 = c2.getSuit();
   if (s1.equals(s2) == true)

   A pair occurs if both the cards have the same value. How would you check this?
   if ((c1.getValue() == c2.getValue())

8. Write a method for the Card class that has the following declaration:
   (notice that this method will return *true* or *false*)

public boolean isHigh()
post: will return true if this Card has a value 8 or larger. Return false otherwise.

```
public boolean isHigh() {
   if (value >=8 )
      return(true) ;
   else
      return(false);
}
```

***Notice that you don't have to use getValue() method!
***You are coding in the Card class and have full access to the member variable value!

9. Write a method for the Card class that has the following declaration:

public boolean isHigher(int val)
post: will return true if this Card has a value higher than the parameter 'val'. Return false otherwise.

A programmer would use the method as follows:

```
if ( c1.isHigher(3) == true)
  System.out.println("Better than a 2 or 3!");
```

```
public boolean isHigher(int val) {
   if (value > val )
      return(true) ;
   else
      return(false);
}
```