NAME: ___________________________________

```
class BeanJar {
public:
    BeanJar(int maxBeans, int beans);
    int getBeans() const;
    bool addBeans(int beans);
    void removeAllBeans();

private:
    int maxBeans;
    int beans;
};
```

Figure 1

The declaration of the *BeanJar* class is given in Figure 1. Instances of the *BeanJar* class represent bean jars of varying capacity. The member variable *maxBeans* represents the capacity of the jar. The member variable *beans* represents the current number of beans in the jar, which must be less than or equal to *maxBeans*. The class provides functions to add beans, remove all beans, and get the number of beans that are currently in the bean jar. The functions to add beans returns a boolean value that indicates whether the operation succeeded or failed. When the addBeans function fails, the number of beans in the jar will not have been changed. For example, if a bean jar has a capacity of 10 beans and there are 9 beans currently in the jar, then *addBeans(2)* will return false and the number of beans in the jar will remain at 9. The *removeAllBeans* function simply sets the number of beans in the jar to zero.

1) Draw the UML class diagram for the *BeanJar* class given in Figure 1. (25 points)

2) Provide an implementation of the *BeanJar* constructor. (25 points)

3) Provide an implementation of the *getBeans* function. (25 points)

4) Provide an implementation of the *removeAllBeans* function. (25 points)

5) Provide an implementation of the *addBeans* function. (25 points)

6) Develop test code to test the *addBeans* function. Make sure the test code provides *good coverage* in the sense that it executes all lines of code in the *addBeans* function. Use the *assert* function in your test code. (25 points)

```
class Integer {
public:
    Integer(int n);
    bool isPrime() const;

private:
    int n;
};
```

Figure 2

The declaration of the *Integer* class is given in Figure 2. Instances of the *Integer* class represent integers. The constructor function takes a single integer argument *n*, which is the value that class instances represent. This value is stored in the member variable, also named *n*. The Integer class also contains a function named *isPrime*. This function returns true if the stored value *n* is prime; otherwise it returns false.

7) Provide an implementation of the *Integer* constructor. (25 points)

8) Provide an implementation of the *isPrime* function. (25 points)

EXTRA CREDIT: Suppose the Integer class contains a function named *primeUpperBound*. This function returns the smallest prime number that is greater than or equal to member variable *n*. Put your answer on the other side of this page. (25 points)