Write the following method that merges two sorted lists into a new sorted list.

```
public static int[] merge(int[] list1, int[] list2)
```

Implement the method in a way that takes at most list1.length + list2.length comparisons. Write a test program that prompts the two sorted lists that are used as input and displays the merged list that created by the method. Note that the first number in the input indicates the number of the elements in the list. This number is not part of the list. Here is a sample run.

```
Enter list1:          5 1 5 16 61 111
Enter list2:          4 2 4 5 6
The merged list is    1 2 4 5 5 6 16 61 111
```

Design a class named Rectangle to represent a rectangle. The class contains:

- Two double data fields named width and height that specify the width and height of the rectangle. The default values are 1 for both width and height.
- A no-arg constructor that creates a default rectangle.
- A constructor that creates a rectangle with the specified width and height.
- A method named getArea() that returns the area of this rectangle.
- A method named getPerimeter() that returns the perimeter.

Draw the UML diagram for the class and then implement the class. Write a test program that creates two Rectangle objects—one with width 4 and height 40 and the other with width 3.5 and height 35.9. Display the width, height, area, and perimeter of each rectangle in this order.

Design a class named Account that contains:

- A private int data field named id for the account (default 0).
- A private double data field named balance for the account (default 0).
- A private double data field named annualInterestRate that stores the current interest rate (default 0). Assume all accounts have the same interest rate.
- A private Date data field named dateCreated that stores the date when the account was created.
- A no-arg constructor that creates a default account.
- A constructor that creates an account with the specified id and initial balance.
- The accessor and mutator methods for id, balance, and annualInterestRate.
- The accessor method for dateCreated.
- A method named getMonthlyInterestRate() that returns the monthly interest rate.
- A method named getMonthlyInterest() that returns the monthly interest.
- A method named withdraw that withdraws a specified amount from the account.
- A method named deposit that deposits a specified amount to the account.

Draw the UML diagram for the class and then implement the class. (Hint: The method getMonthlyInterest() is to return monthly interest, not the interest rate. Monthly interest is balance * monthlyInterestRate. monthlyInterestRate is annualInterestRate / 12. Note that annualInterestRate is a percentage, e.g., like 4.5%. You need to divide it by 100.)

Write a test program that creates an array that consists of 5 Account objects:

- IDs are 1122 5316 1584 4856 9851 respectively.
- Balances are 20,000 54,000 83,000 36,000 4,600 respectively.
- Annual interest rate is 4.5% for all accounts.

Use the withdraw method to withdraw $2,500, use the deposit method to deposit $3,000, and print the balances, the monthly interests, and the dates when each account were created for each account.

Design a class named MyInteger. The class contains:

- An int data field named value that stores the int value represented by this object.
- A constructor that creates a MyInteger object for the specified int value.
- A getter method that returns the int value.
- The methods isEven(), isOdd(), and isPrime() that return true if the value in this object is even, odd, or prime, respectively.
- The static methods isEven(int), isOdd(int), and isPrime(int) that return true if the specified value is even, odd, or prime, respectively.
- The static methods isEven(MyInteger), isOdd(MyInteger), and isPrime(MyInteger) that return true if the specified value is even, odd, or prime, respectively.
- The methods equals(int) and equals(MyInteger) that return true if the value in this object is equal to the specified value.
- A static method parseInt(char[]) that converts an array of numeric characters to an int value.
- A static method parseInt(String) that converts a string into an int value.

Draw the UML diagram for the class and then implement the class. Write a client program that tests all methods in the class.

Design a class named Location for locating a maximal value and its location in a two-dimensional array. The class contains public data fields row, column, and maxValue that store the maximal value and its indices in a two-dimensional array with row and column as int types and maxValue as a double type.

Write the following method that returns the location of the largest element in a two-dimensional array:

```
public static Location locateLargest(double[][] a)
```

The return value is an instance of Location. Write a test program that prompts the user to enter a two-dimensional array and displays the location of the largest element in the array. Here is a sample run:

```
Enter the number of rows and columns in the array: 3 4
Enter the array:
23.5    35      2       10
4.5     3       45      3.5
35      44      5.5     9.6
The location of the largest element is 45 at (1, 2)
```