Solutions to the 2005 exam

Question 1A [4]

In this question, use only the following (**algorithm format**) Boolean expressions:

comparison operators: <, >, =, ≤, ≥, and ≠

Boolean logical operators: NOT, AND, OR

arithmetic operators: +, -, *, /, and MOD (modulo)

variable names and constants.

Use parentheses where necessary. Do **not** use Java syntax!

The Ontario "Drive Clean" program requires that a car must pass an emissions test for licence renewal if both of the following conditions are met:

The car's model year MY is odd and the current year CY is even, or vice versa

The difference between the current year and the model year is at least 3 but no more than 20.

Write a Boolean expression that is true if a car needs a Drive Clean emissions test, and false

Question 1A

```
(((MY MOD 2 = 1) AND (CY MOD 2 = 0)) OR
((MY MOD 2 = 0) AND (CY MOD 2 = 1))) AND
((CY - MY ≥ 3) AND (CY - MY ≤ 20))
```

- Alternative

```
((MY MOD 2 + CY MOD 2) = 1) AND ((CY - MY ≥
  3) AND (CY - MY ≤ 20))
```

Question 1B [4]

What will be printed by the following program? Write your answer below the program.

```
class ABC
{
    public static void main(String[ ] args)
    {
        char[][] x = {{'i', 'j'}, {'4', '5'}};
        char[][] y = {{'x', 'y'}, {'a', 'b'}};
        int i;
        char t;

        for (i = 0; i < 2; i=i+1)
        {
           t = y[(i+1)%2][1];
           y[i][(i+1)%2] = x[i][0];
           y[(i+1)%2][1] = t;
        }

        System.out.print(y[0][1]);
        System.out.println(y[1][0]);
    }
}
```

t = y[1][1] = 'b'

y[0][1] = x[0][0]='i'

y[1][1] = 'b'

t=y[0][1] = 'i'

y[1][0] = x[1][0] = '4'

y[0][1] = 'i'

Question 1B

Answer: i4

Question 1C [4]

```
class C1
{
   private int[] v1 = {1,4,9};
   public int v3;

   public static int m1 (C2 p)
    {
       ...
    }
   private C1 m2 (int m)
    {
       ...
    }
}
```

```
class C2
{
    public static char v3;

    public C2 (int n)
    {
       ...
    }

    private void m3 (int m)
    {
       ...
    }
}
```

Suppose that the following instructions are used in the **main()** method in a class **Test**. Each choice should be considered **independently** – as it if were in its own **main()** method. Circle the letter of the statement which does NOT cause a compilation error.

Question 1C

(a)	int v2 = C1.m1(this);	Error: m1 expects object of type C2 as parameter
(b)	char v3 = C2.v3;	No error
(c)	C1 x = new m2(4);	Error: no constructor m2
(d)	C1 w = new C1(); w.v1[2] = 3;	Error: v1 is private in C1
(e)	C2 y = new C2(); int z = C1.m1(y);	Error: no constructor with zero arguments in C2

Question 2 [8]

```
class AClass
{
  public static void main(String[ ] args)
  {
    aMethod(137210);
    System.out.println( );
  }

 public static void aMethod(int i)
  {

     // see next slide

  }


}
```

- Here is a program that uses recursion.
- What is printed by this program?

Question 2

```
public static void aMethod(int i)
  {
    if (i == 0)
    {  ; // do nothing
    }
    else
    {
      if ( i%10 == 1 )
      {
        System.out.print( "one " );
      }
      else
      {
         if ( i%10 == 2 )
         {
           System.out.print( "two " );
         }
         else
         {
            if ( i%10 == 3 )
            {
              System.out.print( "three " );
            }
            else
            {    ; // do nothing
            }
         }
      }
      aMethod(i/10);
    }
  }
```

Question 2

Answer: one two three one

Question 3 [15]

Translate the following algorithm to a Java method:

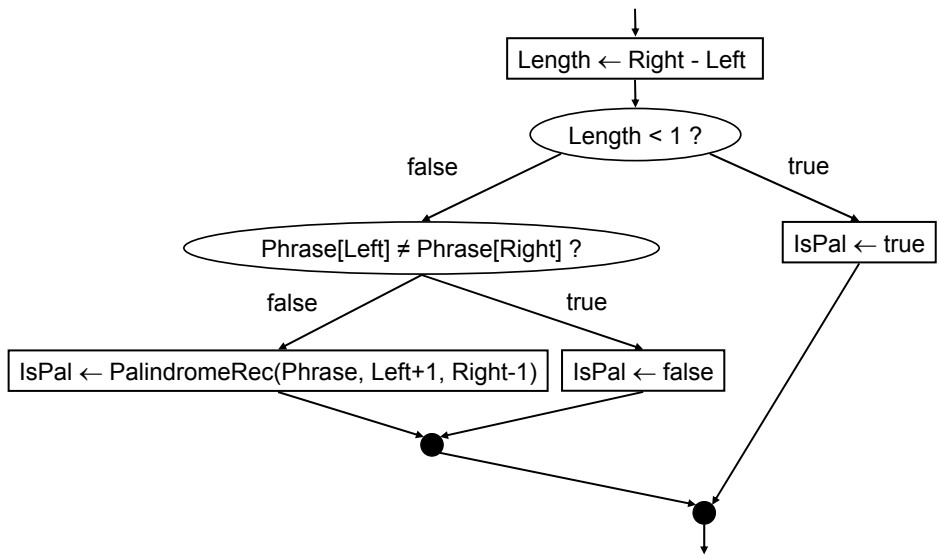
GIVENS:
Phrase: *(an array of characters representing a phrase to be checked)*
Left: *(index of leftmost character to check in Phrase)*
Right: *(index of rightmost character to check in Phrase)*

RESULT:
IsPal *(true if Phrase is a palindrome, and false otherwise)*

INTERMEDIATES:
Length *(length of the interval examined in Phrase)*

HEADER: IsPal ← PalindromeRec(Phrase, Left, Right)

BODY:



Question 3

```
public boolean palindromeRec (char [] phrase, int left, int right)
 {  int length;
      boolean isPal;

      length = right - left;
      if (length < 1)
      {  isPal = true;
      }
      else
      {
         if (phrase[left] != phrase[right])
         {  isPal = false;
         }
         else
         {  isPal = palindromeRec(phrase, left + 1, right - 1);
         }
      }
      return isPal;
 }
```

Question 4 [15]

- In the upcoming election, the candidate who has the most votes in an electoral district wins the riding and becomes the Member of Parliament. However, if the difference in the number of votes between the top two candidates (that is. the candidates with the largest and second-largest numbers of votes) is very close, a recount of the votes will take place.
- Design an algorithm that will take an array Votes containing the number of votes for each of the N candidates in an electoral district, and a value Diff where if the difference in the number of votes for the top two candidates is less than or equal to this value, a recount will take place. The algorithm should return true if the votes should be recounted and false otherwise. The array Votes is **not** sorted.

Answer

GIVENS:

Votes	(array of integers)
N	(length of array)
Diff	(minimum difference in number of votes)

INTERMEDIATES:

First	(the highest number of votes in the array)
Second	(the second highest number of votes in the array)

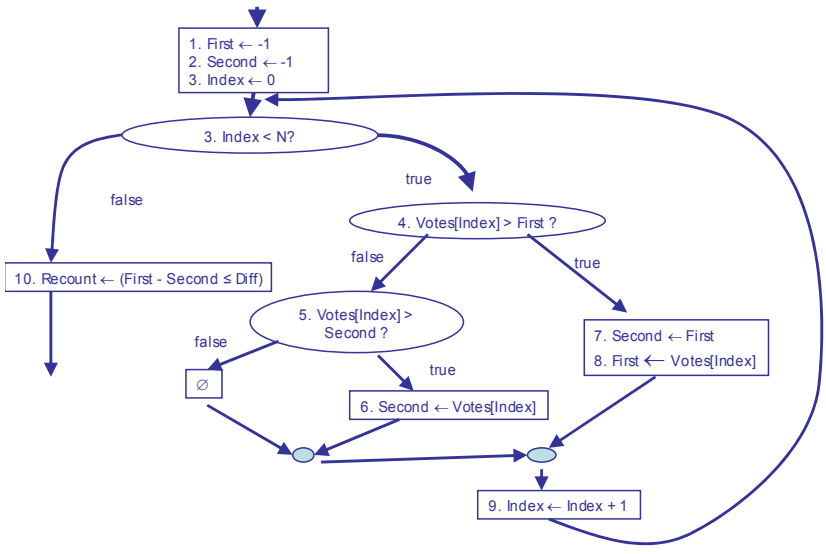
RESULTS:

Recount	(Boolean: true if recount is needed, false otherwise)

HEADER:

Recount ← VotesRecountNeeded(Votes, N, Diff)

BODY:



Question 5 [15]

- A "magic square" is a square matrix of integers where every row and every column can be summed to the same single value. For example, in the matrix ***A*** below, the sum of every row and column is 15.

$$A = \begin{bmatrix} 2 & 9 & 4 \\ 7 & 5 & 3 \\ 6 & 1 & 8 \end{bmatrix}$$

- Write a Java method that will take a square matrix of integers ***A*** and returns true if A represents a "magic square" and false otherwise. Your method should be efficient.

```
public static boolean isMagic( int[][] a )
```

Question 5

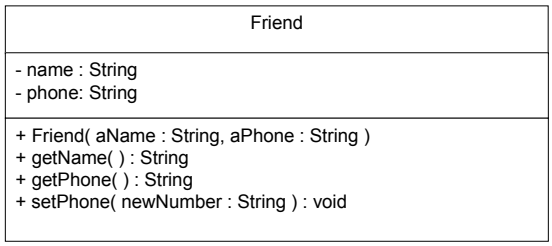
```
public static boolean isMagic( int[][] m )
  {
     int magicSum;
     int sum;
     int row;
     int col;
     boolean magic;

     magic = true;
     magicSum = -1;
     row = 0;
     while ( row < m.length && magic )
     {
        sum = 0;
        col = 0;
        while ( col < m[row].length )
        {
           sum = sum + m[row][col];
           col = col + 1;
        }
```

```
    if ( row == 0 )
        {
           magicSum = sum;
        }
        else
        {
           magic = ( sum == magicSum );
        }
        row = row + 1;
     }
     col = 0;
     while ( col < m[0].length && magic )
     {
        sum = 0;
        row = 0;
        while ( row < m.length )
        {
           sum = sum + m[row][col];
           row = row + 1;
        }
        magic = ( sum == magicSum );
        col = col + 1;
     }
     return magic;
}
```

Question 5

Question 6 [25]

- In this question, you will create a class AddressBook that will be able to store a collection of Friend objects.
- The class Friend, which has already been implemented, provides storage for a person's name and phone number. The class contains a constructor, and two accessor methods. A UML class diagram for the class Friend is as follows:



Question 6

Fill in the methods for AddressBook. Your AddressBook class should provide a constructor and three public methods that would permit the following class TestAddressBook to execute:

```
class TestAddressBook
{
 public static void main (String[] args)
 {
  AddressBook myBook = new AddressBook( 2 );
  myBook.addFriend( new Friend( "Alice", "555-1212" ) );
  myBook.addFriend( new Friend( "Tommy", "555-3434 " ) );
  myBook.addFriend( new Friend( "Pizza", "737-1111" ) );
  myBook.changePhone( "Tommy", "867-5309" );
  myBook.print();
  myBook.changePhone( "Pizza", "310-1010" );
 }
}
```

Question 6

Executing main() would result in the following being printed on the screen :

```
The address book is full.
Name: Alice, Phone: 555-1212
Name: Tommy, Phone: 867-5309
Pizza is not a name in the address book.
```

Complete the class AddressBook

Question 6

```
class AddressBook
{
// ATTRIBUTES: (3 marks)

 Friend [] list;
 int numElements;

// CONSTRUCTOR: (5 marks)
// Takes one integer parameter representing the
// maximum number of entries that can be put into
// the address book.

 public AddressBook(int n)
  {
    list = new Friend[n];
    numElements = 0;
  }
```

Question 6

```
// METHOD addFriend: (6 marks)
// Method parameters: a Friend object that should be added to
// the AddressBook.
// Results:  will print a message if the address book is full
// Modified:  the AddressBook object

public void addFriend( Friend newFriend)
{
  if (numElements < list.length)
  {
    list[numElements] = newFriend;
    numElements ++;
  }
  else
  {
    System.out.println("The address book is full.");
  }
}
```

Question 6

```
// METHOD print: (5 marks)
// Method parameters: (none)
// Returns: (none)
// This method prints a list of names and phone numbers in
// the address book.


public void print()
{
  int index;
  for (index=0; index < numElements; index++)
  {
   System.out.print("Name: " + list[index].getName() + ", ");
   System.out.println("Phone: " + list[index].getPhone());
  }
}
```

Question 6

```
// METHOD changePhone (6 marks)
// Updates the phone number of the first Friend with the
// specified name
// Method parameters: a String which is the name of a Friend,
// and a String which is a new phone number for that Friend.
// Result: an error message if the friend is not in the book

// NOTE:  recall that two strings s1 and s2 can be checked
// for equality with s1.equals(s2)
```

Question 6

```
public void changePhone(String aName, String aPhone)
 {
   int index;
   boolean found = false;
   for (index=0; index < numElements && !found; index++)
   {
     if (aName.equals( list[index].getName()))
     {
       list[index].setPhone(aPhone);
       found = true;
     }
   }
   if (!found)
   {
      System.out.println(aName +
          " is not a valid name in the address book.");
   }
 }

} // End of class AddressBook
```

Question 7 [10]

Write a **recursive** method that takes as a parameter a non-negative integer and generates the following pattern of stars.

You may use a loop to generate one line of stars, but **not** the entire pattern. If the non-negative integer is 4, then the pattern generated is:

```
****
***
**
*
*
**
***
****
```

The header of the method should be the following:

```
public static void stars(int n)
```

Question 7

```
public static void stars( int n )
 {
        int index;
        if ( n > 0 )
        {
            for (index = 0; index < n; index++ )
            {
                System.out.print("*");
            }
            System.out.println( );
            stars( n - 1 );
            for (index = 0; index < n; index++ )
            {
                System.out.print("*");
            }
            System.out.println( );
        }
 }
```