Université d'Ottawa
Faculté de génie

École de science informatique
et de génie électrique

uOttawa
L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical Engineering
and Computer Science

ITI 1120/1520 Fall 2011
Introduction to Computing I

FINAL EXAMINATION

Length of Examination: 3 hours **December 22, 2011, 14:00**
Professors: Diana Inkpen, Nathalie Japkowicz

Identification: (complete this section using ink)

Family name : _____

Given names : _____

Student number : _____ Signature :_____

Instructions: Please read carefully!

1. This is a closed-book test. **No books, papers, calculators or other electronic devices are permitted.**
2. Answer all questions on the question sheet in the area provided**.**
3. **Les réponses en français sont acceptées**.
4. Algorithms are to be described using the format from the lectures and the notes.
5. You can use the back of the question sheet pages, or page 16, for calculations and other work. Pages 16 and 17 can be detached as they will not be marked.
6. If a question appears to be unclear, state a hypothesis and proceed to answer the question.
7. No student will be allowed to leave the exam room during the last 15 minutes of the exam.
8. When time is called at the end of the exam, **turn your paper over immediately. Stay in your place, and remain quiet until the proctors have collected all papers** and give clearance to leave.

1 (12)	2 (8)	3 (15)	4 (15)	5 (15)	6 (25)	7 (10)	Tot. (100)

Question 1A) (4 marks)

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!

Suppose that a variable **P** is true if the pricing of a store's items has not been changed in three months and false if it has. Suppose as well that a variable **G** is true if the economy is good and false, otherwise. The pricing of a store's items needs to be changed **when at least one of the following conditions applies**:

- The pricing of a store's items has not been changed in three months and the difference between the acceptable and current average prices of luxury items (a variable called **D)** is greater than 150, or the difference between the acceptable and current average prices of regular items (a variable called **R)** is greater than 50 and the number of regular items in stock **N** is at least 500.
- The number of days to Christmas **C** is smaller than 35, or it is between 35 and 55 (both included) and the economy is bad.

Write a Boolean expression that is true if the pricing of a store's items needs to be changed and false otherwise.

Answer:

((P and D > 150) OR (R > 50 AND N > 500)) OR ((C < 35) OR ((C >= 35 AND C <= 55) AND (NOT G)))

Question 1B) (4 marks)

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

```
class Q1B
{   public static void main(String[] args)
    {
       int[][] x = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
       int i, j;

       for (i = 0; i < 3; i=i+1)
         for (j = 2; j >= 0; j=j-1)
             x[i][j] = x[j][i]-x[i][j];

       for (i = 0; i < 3; i=i+1)
        { for (j = 0; j < 3; j=j+1)
                System.out.print(x[i][j] + " ");
          System.out.println();
        }
    }
}
```

Answer:

```
 0  2  4
-2  0  2
-3 -6  0
```

Question 1C) (4 marks)

Consider the following two classes:

```
class C1
{
 private int v1;
 private static int v2;

 public void setV1 (int val)
  {
    v1 = val;
  }

 public static void setV2(int val)
  {
    v2 = val;
  }
}
```

```
class C2
{
  public static char[] v3={'a','b'};
  private int v4;

  public C2 (int n)
   {
     v4 = n;
   }

  public void m1 (C1 x)
   {
      x.setV1(5);
   }
}
```

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 <u>only</u> statement that <u>does not</u> cause an error.**

a)
```
C1 x = new C1();
C1.setV1(6);
```

b)
```
C2.v3[1] = "c".charAt(0);
```

c)
```
C2 z = new C2(10);
z.v4 = 15;
```

d)
```
int x = new C2(3);
C1.setV2(x.v3.length);
```

e)
```
C1 y = new C1();
C2.m1(y);
```

Answer : b)

Question 2: (8 marks)

Here is a program that uses multi-dimensional arrays.

```
public class Q2
{
  public static void main( String[] args )
  {
    int[][] m = new int[4][4];

    fill_matrix(m);

    for (int i = 0; i < m.length; i = i + 1)
    {
      for (int j = 0; j < m[i].length; j = j + 1)
      {
        System.out.print (m[i][j] + " ");
      }
      System.out.println();
    }
  }

  public static void fill_matrix(int [][] p)
  {
    //populate matrix
    for (int i = 0; i < p.length; i = i + 1)
    {
      for (int j = 0; j < p[i].length; j = j + 1)
      {
        if (i == p.length - 1 - j)
        {
           p[i][j] = 1;
        }
        else
        {
          p[i][j] = 0 ;
        }
      }
    }
  }
}
```

What is printed during the execution of the method `main` in this class?

Answer :

0 0 0 1
0 0 1 0
0 1 0 0
1 0 0 0

Question 3A): (10 marks)

Translate the algorithm from Page 17 into a **Java method**.

Answer:

```
public static int recMethod (int[] x, int n)
{
    int result; // RESULT
    int s; // INTERMEDIATE

    if (n < 0)
    {
      result = 1;
    }
    else
    {
      s = recMethod(x, n-1);

      if (x[n] % 2 != 0)
      {
        result = s * x[n];
      }
      else
      {
        result = s;
      }
    }
    return result;

}
```

Question 3B) (5 marks)

What does the following main method print? This method is in the same class as your recMethod.

```
public static void main(String[ ] args)
{
      int [] a = {1, -5, -4, 8, -3, 2};
      int result = recMethod(a, a.length - 1);
      System.out.println(result);
}
```

Answer:

15

Question 4: (15 marks)

Design an algorithm that will take an array A of length N and produce a new array B that contains the elements stored at the even index positions (0, 2, ...) of array A.
For example, if A = {5, 4, -1, 0, 6}, then B = {5,-1,6}. If A={1,2,3,4,5,6,7}, then B={1,3,5,7}.
Make sure that the array B has the right size when you allocate memory for it. You can use the predefined method makeNewArray(s: integer) to create a new array of size s.

[Use this page and the next for the answer; it is suggested to put the body on the next page.]

Note: in this problem, we are not looking for Java code, but rather for an answer in algorithmic format. A Java answer will not receive any points.

Answer:

GIVENS:
A (array of integers)
N (number of elements in the array)

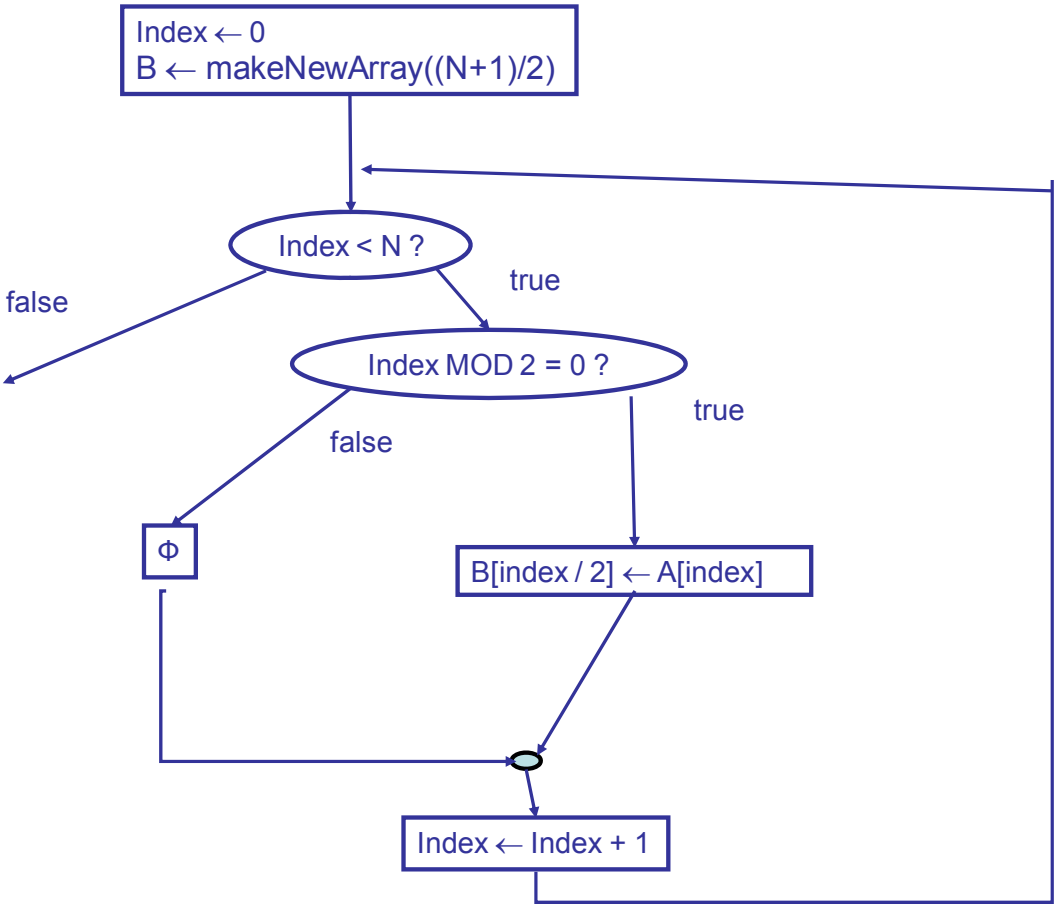
INTERMEDIATES:
Index (index for the array)

RESULT: B (new array, with half the elements of the array A)

HEADER: B ← buildNewArray (A, N)

BODY:

Question 4 continued:



Question 5: (15 marks)

Write a Java method that takes a matrix of integers ***A*** and adds the product of each row together, except when a row contains zeros's, in which case the product of all the row elements but the zeros is added.

NB: 1) If all the row elements, but one, are zeros then the non-zero element represents the product of the row; 2) If all the row elements are zeros, then 0 represents the product of the row.

For example, the method should return 257 for the following matrix

$$A = \begin{vmatrix} 2 & 9 & 4 & 2 \\ 0 & 5 & 7 & 3 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

because

- The product (as defined) of the elements of the first row is 144
- The product (as defined) of the elements of the second row is 105
- The product (as defined) of the elements of the third row is 8
- The product (as defined) of the elements of the fourth row is 0
- 144 + 105 + 8 + 0 = 257

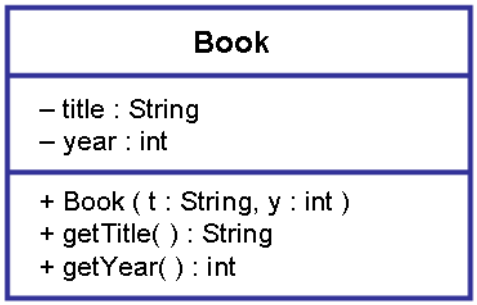
Answer:

```
public static int sumOfRowProducts (int[] [] a)
  {
    int sum =  0;
    int product, zeros;

    for (int row = 0; row < a.length; row++)
    {
      product = 1;
      zeros = 0;
      for (int col = 0; col < a[row].length; col++)
      {
          if (a[row][col] != 0)
          {
            product = product * a[row][col];
          }
          else
          {
            zeros = zeros + 1; // counts how many zeros are in a row
          }

          if (zeros == a[row].length)
          {
            product = 0;
          }
          else
          {
            ; // do nothing
          }
      }
      sum = sum + product;
    }
    return sum;
 }
```

Question 6: (25 marks)

In this question, you will create a class Journal (for a library) which contains a collection of Loan objects, where each Loan object contains a Book object.

The class Book, which has already been implemented, stores the title of the book and the year it was published. There is a constructor which sets these two values, and two accessor methods to obtain the values. A UML diagram for this class appears below:



The class `Loan`, which is also already implemented, stores a `Book` object which has been borrowed from the library. The `Loan` stores a reference to the book that is borrowed, a `String` with the name of the borrower, and a `boolean` variable that indicates whether the `Book` has been returned or not. The `Book` and the borrower's name are set by the constructor, and initially the `Book` is not returned. There are accessors for each of the three values stored in the book. In addition, there is a method to be called when a book is returned, that sets the Boolean variable to be true. Java code for this class is below:

```
class Loan
{
   private Book aBook;
   private String borrowerName;
   private boolean returned;

   public Loan(Book b, String n)
   {
      this.aBook = b;
      this.borrowerName = n;
      this.returned = false;
   }
   public Book getBook()
   {
      return this.aBook;
   }
   public String getBorrower()
   {
      return this.borrowerName;
   }
   public boolean isReturned()
   {
      return this.returned;
   }
   public void returnBook()
   {
      this.returned = true;
   }
}
```

Question 6:(continued)

On the following pages, complete the class `Journal`, and the five public methods that would permit the `main` method in the class `TestJournal` to execute:

```
class TestJournal
{
  public static void main(String[] args)
  {
    boolean r; // result

    Journal.init(3);
    r = Journal.borrow( new Book("Harry Potter VI", 2005), "Daniel");
    r = Journal.borrow( new Book("Da Vinci Code", 2003), "Diana");
    r = Journal.borrowNewBook("Java for Dummies", 2001, "Daniel");
    System.out.println("Loan successful: " + r);
                                    // true, because the journal was empty

    r = Journal.borrowNewBook("Tintin au Congo", 1946, "Diana");
    System.out.println("Loan successful: " + r); // Journal is full
    Journal.init(4); // The Journal cannot be initialized again

    Journal.returnBook("Da Vinci Code", "Diana"); // 1 book returned
    Journal.printLoans(); // There are 2 books still on loan
  }
}
```

This will print the following output:

```
Loan successful: true
Loan successful: false
Sorry, size already initialized to 3

Books on loan:
Harry Potter VI borrowed by Daniel
Java for Dummies borrowed by Daniel
```

Complete the class Journal (which begins below) on this and the following pages.

```
class Journal
{
      // ATTRIBUTES (These are provided for you.)

       private static Loan [] loans;  // Array of loans

       private static int numLoans; // Keeps track of how much of the array
                                    // is used (the number of actual loans)

       // Indicates if the size of the Journal has already been set

       private static boolean initialized = false;
```

Question 6:(continued)

// CLASS METHOD **`init`** *: (5 marks)*
// Parameters: an integer representing the maximum number of loans that can be contained
// in the journal.
// Result: prints a message if the journal has already been initialized (see page 10 for format).

```
public static void init( int size )
   {
      if ( initialized )
      {
         System.out.println( "Sorry, size already initialized to "
               + loans.length );
      }
      else
      {
         loans = new Loan[size];
         numLoans = 0;
         initialized = true;
      }
   }
```

// CLASS METHOD **`borrow`** *(5 marks)*
// Parameters: a `Book` *object and the name of a borrower (a* `String`*).*
// Result: Returns true if there is space in the `Journal` *for the new* `Loan`*, and false otherwise.*

```
public static boolean borrow( Book aBook, String borrowerName )
   {
      boolean result;
      if ( numLoans == loans.length )
      {
         result = false;
      }
      else
      {
         Loan newLoan = new Loan( aBook, borrowerName );
         loans[numLoans] = newLoan;
         numLoans = numLoans + 1;
         result = true;
      }
      return result;
   }
```

Question 6:(continued)

// CLASS METHOD **`borrowNewBook`** *(5 marks)*
// Parameters: the title and year of a new book, and the name of its borrower
// Result: Returns true if the loan was successfully added to the Journal, and false otherwise.
// Note: this method must use the **`borrow`** *method defined on the previous page.*

```
public static boolean borrowNewBook( String bookTitle, int year,
                                     String borrowerName )
   {
      return Journal.borrow( new Book( bookTitle, year ), borrowerName );
   }
```

// CLASS METHOD **`returnBook`** *(5 marks)*
// Parameters: the title of a book, and the name of a borrower (both are `String` *variables).*
// This method locates a `Loan` *in the* `Journal` *based on the name of the* `Book` *and the borrower's name*
// and sets the returned variable in the corresponding `Loan` *object to true.*
// NOTE: recall that two strings `s1` *and* `s2` *can be checked for equality with* `s1.equals(s2)`

```
public static void returnBook( String bookTitle, String borrowerName )
   {
      int index;
      boolean found;

      index = 0;
      found = false;
      while ( !found && index < numLoans )
      {
         Loan aLoan = loans[index];
         Book aBook = aLoan.getBook( );
         if ( bookTitle.equals( aBook.getTitle( ) ) )
         {
            aLoan.returnBook( );
            found = true;
         }
         else
         {
            index = index + 1;
         }
      }

   }
```

Question 6:(continued)

// CLASS METHOD **`printLoans`** *(5 marks)*
// Prints the title and the borrower of all books in the Journal that have not yet been returned.

```
   public static void printLoans( )
   {
      int index;  // Index for the loans in the journal.

      System.out.println( );
      System.out.println("Books on loan:");

      for ( index = 0; index < numLoans; index = index + 1 )
      {

        Loan theLoan = loans[index];

        if ( !theLoan.isReturned( ) )
        {
           System.out.print( theLoan.getBook( ).getTitle( ) );
           System.out.print( " borrowed by " );
           System.out.println( theLoan.getBorrower( ) );
        }









       }

   }
} // End of class Journal
```

Question 7: (10 marks)

A palindrome reads identically from left to right and from right to left. For example, {1, 4, 5, 6, 7, 6, 5, 4, 1} is a palindrome, but {1, 4, 6, 6, 7, 6, 5, 4, 1} is not. In this problem, we define the notion of a funny palindrome which is an **odd** array of integers with at least 3 elements, which is such that for all integers in the array, except for the one in the middle:

- either the two integers located at opposite locations are identical (like in a regular palindrome), or
- the sum of two consecutive integers is identical to the sum of the two consecutive integers located at opposite locations.

For example, a = {1, 2, 5, 6, 4, 5, 3, 7, 4, 3, 1} is a funny palindrome since

- a[0] = a[10] = 1, and
- (a[1] + a[2]) = (a[9] + a[8]) = 7, and
- (a[3] + a[4]) = (a[7] + a[6]) = 10, and
- a contains an odd number of integers.

Write a **recursive** Java method that verifies whether an array containing an odd number of integers is a funny palindrome. You can assume that the array passed on to the method always has an odd number of integers. Your method should have the following header, and it will be called from a main method as funnyPal(a, 0, a.length-1).

Answer:

```
public static boolean funnyPal (int[] a, int low, int high)
{
    boolean result;
    if (low == high)
    {
      result= true;
    }
    else
    {
      if (a[low]==a[high])
      {
        result = funnyPal(a, low+1, high-1);
      }
      else
      {
        if ((a[low] + a[low+1]) == (a[high] + a[high-1]))
        {
           result = funnyPal(a, low+2, high-2);
        }
        else
        {
         result = false;
        }
      }
    }
    return result;

 }
```

This page is for calculations and other work, and can be detached. This page will not be marked.

Algorithm for Question 3 (this page can be detached)

GIVENS:	X	(an array)
	N	(the last index in the array)
INTERMEDIATES:	S	(result for array up to index N-1)
RESULT:	Result	(result for array up to last index N)
HEADER:	Result ← RecMethod(X, N)	
BODY: