German University in Cairo
Computer Science and Engineering Department
Prof. Dr. Slim Abdennadher

May 8, 2008

Introduction to Computer Programming
Spring term 2008
Midterm Exam

Bar Code

Instructions: Read carefully before proceeding.

1) Duration of the exam: 2 hours (120 minutes).
2) (Non-programmable) Calculators are allowed.
3) No books or other aids are permitted for this test.
4) This exam booklet contains 13 pages, including this one. Four extra sheets of scratch paper are attached and have to be kept attached. **Note that if one or more pages are missing, you will lose their points. Thus, you must check that your exam booklet is complete**.
5) Write your solutions in the space provided. If you need more space, write on the back of the sheet containing the problem or on the three extra sheets and make an arrow indicating that. **Scratch sheets will not be graded unless an arrow on the problem page indicates that the solution extends to the scratch sheets**.
6) When you are told that time is up, stop working on the test.

Good Luck!

Don't write anything below `;-)`

Exercise	1	2	3	4	5	6	7	8	$\sum$
Possible Marks	6	8	6	6	8	6	4	8	52
Final Marks									

Exercise 1 (6 Marks)

a) Rewrite the following switch statement by using if-then-else statements:

```
switch (i)
{
 case 4: x = 0;
 case 5:
 case 6:
 case 7: j = 1; break;
 case 8: j = 2;
}
```

Solution:

```
if (i == 5 || i == 6 || i == 7)
   j = 1;
else if (i == 4)
   {x = 0; j = 1;}
     else if (i == 8)
          j = 2;
```

b) Simplify the following program fragment, taking out unnecessary comparisons. Assume that `age` is an `int` variable.

```
if (age > 64)
   System.out.println("Senior voter");
if (age < 18)
   System.out.println("Under age");
if (age >= 18 && age < 65)
   System.out.println("Regular voter");
```

Solution:

```
if (age >= 18 && age < 65)
   System.out.println("Regular voter");
else if (age < 18)
       System.out.println("Under age");
     else
       System.out.println("Senior voter");
```

Exercise 2 (8 Marks)

Write a Java program using a switch statement. The program inputs a single letter and outputs the corresponding digit on the telephone. The letters and digits on a telephone are grouped as follows:

2 = ABC 3 = DEF 4 = GHI 5 = JKL
6 = MNO 7 = PRS 8 = TUV 9 = WXY

No digit corresponds to either Q or Z. For these two letters, your program should print a message indicating that they are not used on a telephone. The program might operate as follows:

```
Enter a single letter, and I will tell you what the corresponding digit is on the telephone.
R
The digit 7 corresponds to the letter R on the telephone.
```

Here is another example

```
Enter a single letter, and I will tell you what the corresponding digit is on the telephone.
Q
There is no digit on the telephone that corresponds to Q.
```

Your program should print a message indicating that there is no matching digit for any nonalphabetic character and lowercase letters.

Complete the following program:

```
import java.io.*;

public class Phone {

   public static void main(String[] args) throws IOException {

        InputStreamReader instream= new InputStreamReader(System.in);
        BufferedReader stdin= new BufferedReader(instream);
        String s;
        System.out.println("Enter a single letter, and I will tell
                           you what the corresponding digit is on the telephone.");
```

Solution:

```
import java.io.*;

public class Phone {

   public static void main(String[] args) throws IOException {

        InputStreamReader instream= new InputStreamReader(System.in);
        BufferedReader stdin= new BufferedReader(instream);
        String s;
        System.out.println("Enter a single letter, and I will tell
                           you what the corresponding digit is on the telephone.");
        s = stdin.readLine();
        char c = s.charAt(0);

        switch (c)
        {
         case 'A': case 'B': case 'C': System.out.println("The digit " + 2 +
                                                          " corresponds to " + c); break;
         case 'D': case 'E': case 'F': System.out.println("The digit " + 3 +
                                                          " corresponds to " + c); break;
         case 'G': case 'H': case 'I': System.out.println("The digit " + 4 +
```

```
                                                        " corresponds to " + c); break;
         case 'J': case 'K': case 'L': System.out.println("The digit " + 5 +
                                                        " corresponds to " + c); break;
         case 'M': case 'N': case 'O': System.out.println("The digit " + 6 +
                                                        " corresponds to " + c); break;
         case 'P': case 'R': case 'S': System.out.println("The digit " + 7 +
                                                        " corresponds to " + c); break;
         case 'T': case 'U': case 'V': System.out.println("The digit " + 8 +
                                                        " corresponds to " + c);
         case 'W': case 'X': case 'Y': System.out.println("The digit " + 9 +
                                                        " corresponds to " + c);
         case 'Q': case 'Z': System.out.println(" There is no digit on the telephone
                                                that corresponds to " + c); break;
        default:  System.out.println("Not a valid entry");
        }

    }

}
```

Exercise 3 (6 Marks)

Often, in dictionary lookups and word-processors, we are interested in finding a string which is the closest match to another. Let us define the degree of match as the number of positions in which the two strings have the same character. Write a method which will accept two strings as arguments and return the number of positions in which they are identical. For example, if your method is named `howClose`, then

```
howClose("abcdef","acbdef")
```

should return 4 since positions, 0, 3, 4 and 5 are the same. If the strings are of different length then the matching should only go up to the last position in the shorter of the two. That is, `howClose("abcd","acbdef")` should return 2.

Solution:

```
public class HowClose {

    public static int howClose(String word1, String word2)
    {

        int count = 0;
        int len;
        if (word1.length() >= word2.length())
            len = word2.length();
        else
            len =  word1.length();



        for (int i = 0; i < len; i++)
        {
            if (word1.charAt(i) == word2.charAt(i))
               count ++;
        }

        return count;

    }


    public static void main(String[] args) {
          System.out.println(howClose("abcd", "acbdefa"));
    }

}
```

Exercise 4 (6 Marks)

Write a Java method which accepts a string of positive digits, and returns a string of all the positive digits that are NOT found in the argument string. For example, if your method is named missing, then `missing("33125479")` would return `"68"`.

Hint: Please note that characters can be compared in different ways. One way is to use the unicode values of the characters, `'0'`, `1`, ..., `9`. Unicode of `'0'` is `48`, ..., Unicode of `'9'` is `57`.

Solution:

```
public class Missing {

    public static String missing(String word)
    {
        String result = "";
        boolean found;
        for (int i = 1; i <= 9; i++)
        {
            found = false;
        for (int j = 0; j < word.length(); j++)
        {
            if (word.charAt(j) == 48 + i)
               found = true;
        }
        if (!found)
           result += i;
        }
        return result;
    }


    public static void main(String[] args) {

       System.out.println(missing("125678888876"));
    }

}
```

Exercise 5 (8 Marks)

Write a Java program that computes the prime factorization of a given integer number `n`. Prime Factorization is finding which prime numbers you need to multiply together to get the original number.

Examples of different runs of the program

```
The prime factorization of 81 is: 3 3 3 3

The prime factorization of 168 is: 2 2 2 3 7

The prime factorization of 4444444444 is: 2 2 11 41 271 9091
```

Solution:

```
public class Factors {

    public static void main(String[] args) {

        // command-line argument
        long n = 81;

        System.out.print("The prime factorization of " + n + " is: ");

        // for each potential factor i
        for (long i = 2; i <= n / i; i++) {

            // if i is a factor of N, repeatedly divide it out
            while (n % i == 0) {
                System.out.print(i + " ");
                n = n / i;
            }
        }

        // if biggest factor occurs only once, n > 1
        if (n > 1) System.out.println(n);
        else       System.out.println();
    }
}
```

Exercise 6 (6 Marks)

a) What is output of the following program? Justify your answer.

```
void myfunc(int i) {i = 6;}

public static void main(String [] args)
{
   int i = 4;
   myfunc(i);
   System.out.println(i);
}
```

Solution:

4: call by value.

Since there was a typo in the exam `my fuc`, students answered the question as follows: The program does not compile. The solution will be considered correct.

b) What is the output of the following program? Justify your answer.

```
public class recursion {
  public static void dude (int y)
  {
    if (y == 0)
    {
      System.out.println("y = " + y + " :Statement inside the if");
      dude(y + 1);
    }

    if (y == 1)
    {
      System.out.println("y = " + y + " :Statement inside the if");
      dude(y + 1);
    } else
    {
       System.out.println("y = " + y + " :Statement outside the if");
    }
   }

  public static void main (String [] args)
  {
   dude(0);
  }
}
```

Solution:

```
y = 0 :Statement inside the if
y = 1 :Statement inside the if
y = 2 :Statement outside the if
y = 0 :Statement outside the if
```

Exercise 7 (4 Marks)

Given the following program:

```
public static void mystery1(int a, int b) {
    if (a <= b) {
        int m = (a + b) / 2;
        System.out.print(m + " ");
        mystery1(a, m-1);
        mystery1(m+1, b);
    }
}
```

What is the output of the main method? Justify your answer with a tracing table.

```
public static void main(String[] args) {
    int N = 6;
    mystery1(0, N);
}
```

Solution:

```
3 1 0 2 5 4 6
```

Exercise 8 (8 Marks)

a) Write and test a recursive method `loanLength` that takes a loan amount and an annual interest rate and a monthly payment. The method should return the number of months until the loan is paid off, assuming that the interest is compounded monthly.

For example:

- `loanLength(1000, 0.10, 200)` is 6 months
- `loanLength(1000, 0.10, 1050)` is 1 month
- `loanLength(0, 0.90, 50)` is 0 months

In the last month, the payment may be less than the monthly payment amount. So, think of the loan as being paid off when the loan amount is 0 or less.

Solution:

```
public class Loanlength {

    public static int loanLength(double a, double i, double r)
    {

        if (a <= 0)  return 0;
        else
        { a = (a-r) + (a-r) * (i/12);
          return 1 + loanLength(a, i,r);
        }

    }

    public static void main(String[] args) {
       System.out.println(loanLength(1000,0.1,200));
    }

}
```

b) Enhance your previous method with additional methods to print out the loan amount remaining at each month. Each line of the output should show the number of months that have gone by and the amount of principle remaining, to the nearest integer. For example, for a 1000 loan with an interest rate of 10% and a monthly payment of 250, the printed output would be as follows:

```
Month 0: 1000.0
Month 1: 756.25
Month 2: 510.46875
Month 3: 262.6393229166667
Month 4: 12.744650607638908
Month 5: -239.23247730396412
```

No Loops are allowed. ONLY RECURSION

This question is a bonus question and it is worth 6 marks!

Solution:

```
public class Loanlength {
    public static int loanLength(double a, double i, double r)
    {

        if (a <= 0)  return 0;
        else
```

```
        {
          a = (a-r) + (a-r) * (i/12);
          return 1 + loanLength(a, i,r);
        }

    }


    public static void loan(double a, double i, double r)
    {
        int x = loanLength(a,i,r);
        loan(a,i,r,x,0);
    }

    public static void loan(double a, double i, double r, int x, int c)
    {
        if (x == c)
           System.out.println("Month " + c + ": " + a);
        else
            {
              System.out.println("Month " + c + ": " + a);
              a = (a-r) + (a-r) * (i/12);
              loan(a,i,r,x,c+1);
    }
    }

    public static void main(String[] args) {

       loan(1000,0.1,250);


    }

}
```

Extra Sheet

Extra Sheet

Extra Sheet

Extra Sheet