German University in Cairo
Media Engineering and Technology Faculty
Prof. Dr. Slim Abdennadher
Assoc. Prof. Milad Ghantous
Dr. Mohamed Hamed

April 18, 2023

CSEN 202: Introduction to Computer programming
Spring Semester 2023

Midterm Exam

Bar Code

Instructions: Read carefully before proceeding.

1) Duration of the exam: 2 hours (120 minutes).
2) No books or other aids are permitted for this test.
3) This exam booklet contains 11 pages, including this one. Three 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**.
4) 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**.
5) 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	$\sum$
Possible Marks	12	12	12	12	15	8	71
Final Marks							

Exercise 1 (12 Marks)

Given the following Java programs/expressions, write the output after the execution and the type of the output. **If a program does not compile or causes an exception at runtime, write error and mention its type.**

a)
```
long x = 10;
int y = 5;
y = y*x;
System.out.println(y);
```

Solution:

```
error. Cannot convert from long to int      #2 marks
```

b)
```
short x = 6;
float y = 3;
double z = 2;
System.out.println(x*y/z);
```

Solution:

```
9.0 (double)                                 #2 marks
```

c)
```
int i = 1, j = 10;
do
{
    if(i++ > --j)
    {

    }
} while (i < 5);
System.out.println("i = " + i + "and j = " + j);
```

Solution:

```
i = 5 and j = 6  (Integers)                  #2 marks
```

d)
```
int i1 = 5;
int i2 = 6;
String s1 = "7";
System.out.println(i1 + i2 + s1);
```

Solution:

```
"117"  String                                #2 marks
```

e)
```
byte x;
int a = 270;
double b = 128.128;
x = (byte) a;
System.out.println("a and x " + a + " " + x);
a = (int) b;
System.out.println("b and a " + b + " " + a);
x = b;
System.out.println("b and x " + b + " " + x);
```
State which statements will work and write their output, and which will give an error (if any).

Solution:

```
a and x 270 14 \\
b and a 128.128 128 \\
Third one error, cant covert double to byte       #2 marks
```

f) Assume that a, b and c, are boolean variables, what is the output b?

```
b = (a || !c) == (c && !a)
```

Solution:

```
b will always be false (boolean)                     #2 marks
```

Exercise 2 (12 Marks)

Write a Java program that takes the day, month and the year from the user and computes the date of the next day, and prints it. Assume that all inputs are in integer form.
Note: The months that have 30 days are: April, June, September and November. February has 28 days and the rest of the months have 31 days.

Examples:

```
Day = 1
Month = 1
Year = 2022
Output: 2/1/2022

Day = 30
Month = 5
Year = 2022
Output: 1/6/2022

Day = 31
Month = 12
Year = 2022
Output: 1/1/2023
```

Solution:

grading:

if student covers all cases, in different ways than the solution, he/she gets full grade.
Input: 1 pt
next month 31 days: 2 pts
next month 30 days: 2 pts
February: 2pts
next year: 2 pts
regular next day: 2 pts
output: 1 pt

```
import java.util.Scanner;
class Main {
  public static void main(String args[]) {
    Scanner sc = new Scanner(System.in);
    int day = sc.nextInt();
    int month = sc.nextInt();
    int year = sc.nextInt();
    int nextDay = day, nextMonth = month, nextYear = year;
    if(day == 31 & (month == 1 || month == 3 ||month == 5 ||month == 7
    ||month == 8 ||month == 10 ||month == 12))
    {
      nextDay = 1;
      nextMonth = month+1;
      if(month==12)
      {
        nextMonth = 1;
        nextYear= year + 1;
      }
    }
    else
    {
      if((day == 28 && month == 2) || (day == 30 && (month == 4 || month ==
      6 ||month == 9 ||month == 11)))
      {
        nextDay = 1;
        nextMonth = month+1;
      }
      else
        nextDay = day +=1;
    }
    System.out.println(nextDay + "␣" + nextMonth+"␣" + nextYear);
  }
}
```

Other solution:

```
import java.util.Scanner;
public class Main {
public static void main(String[] args) {
Scanner sc = new Scanner(System.in);
System.out.println("Enter␣the␣day");
int day = sc.nextInt();
System.out.println("Enter␣the␣Month");
int month = sc.nextInt();
System.out.println("Enter␣the␣year");
int year = sc.nextInt();
switch(month)
{
    case 4: case 6: case 9: case 11:
            if(day==30)
            {
                month++;
                day = 1;
            }
            else
                day++;
            break;
    case 1: case 3: case 5: case 7: case 8: case 10:
            if(day==31)
            {
                month++;
                day = 1;
            }
            else
                day++;
            break;
    case 2:
            if(day == 28)
            {
                month++;
                day = 1;
            }
            else
                day++;
            break;
    case 12:
            if(day == 31)
            {
                year++;
                month = 1;
                day = 1;
            }
            else
                day++;
            break;
    default:
            System.out.println("Invalid␣inputs");
}
System.out.println("output:␣" + day +"/" + month + "/" + year);
}
}
```

Exercise 3 (12 Marks)

Write a Java method `sentenceReverse`, that takes a string as a parameter and reverses the order of the words in the given string, then return it.

Examples:

```
sentenceReverse("I like the CSEN202 course") returns "course CSEN202 the like I"

sentenceReverse("Hello") returns "Hello"
```

Solution:

Grading scheme:
Method signature: 1 pt
input and initialization: 1 pt
for loop: 1.5 pts
detecting space(end of word): 2 pts
concatenation of word: 1.5 pt
concatenation of sentence: 1.5 pt
resetting word: 1pt
handling last word: 1.5 pt
return: 1 pt

```
public static String sentenceReverse(String s)
{
    String s2 = "";
    String out = "";
    for (int i =0; i<s.length();i++)
    {
      if (s.charAt(i) != ' ')
         s2 += s.charAt(i);
        else
        {
            out = s2 + " " + out;
            s2 = "";
        }
    }
      out = s2 + " " + out;
      System.out.println(out);
}
```

Other solution:

```
public static String sentenceReverse(String s)
{
    for(int i=0; i<s.length();i++)
    {
        if(i==s.length()-1) //last character
        {
            word += s.charAt(i);
            rev = word + " " + rev;
        }
        else  // not last character
            if(s.charAt(i)!=' ')
              word += s.charAt(i);
            else
            {
                rev = word + " " + rev;
                word="";
            }
    }
}
```

Exercise 4 (12 Marks)

Write a Java method `findMax`, that takes as input a positive integer and finds the maximum value that can be obtained from the input number by performing operations on its digits, and prints it. The operations that can be performed are addition and multiplication only.
Note: You don't have to check every possible permutation (arrangement) of the digits. You just need to traverse the digits only once.
You are **not** allowed to convert the number into Strings. You have to manipulate the integers.

Examples:

```
findMax(891) prints 80
Explanation: 1 + 9 > 9*1 => 10
and 10*8 > 10+8, thus 80 is the maximum.

findMax(12310) prints 9
Explanation: 0 + 1 > 0 *1 => 1
and 1 + 3 > 1 * 3
then 4 * 2 > 4 + 2
then 8 + 1 > 8 * 1, thus 9 is the maximum

findMax(-6590) prints "The number needs to be positive"
```

Solution:

Grading scheme:
Method signature: 2 pts (void: 1 pt, parameters: 1pt)
initialising max: 1 pt
while: 1 pt
condition add/mul: 3 pts
adjusting the max: 2 pts
n/10: 1 pt
check if negative: 1pt
printing: 1

```
public static void findMax(int n)
{
    if (n < 0)
        System.out.println("The number needs to be positive");
    else
      {
        int res = n%10;
        n = n/10;
        while (n != 0)
        {
          int x = n%10;
          if (x == 0 || x == 1 || res < 2 )
              res += x;
          else
              res *= x;
          n = n/10;
        }
        System.out.println(res);
      }
  }
}
```

Other solution:

```
public static void findMax(int n)
{
    int max=0;
    int d1 =0;

        while(n!=0)
        {
            d1 = n%10;

            if(max+d1 > max*d1)
              max=max+d1;
            else
              max= max*d1;

            n= n/10;
  }
}
```

Exercise 5 (15 Marks)

Given a string `s`, write a Java method `findNonRepeat` that finds the `first` non-repeating character in `s` and returns its index. If it does not exist, you should return -1.
Note: You are only allowed to use string methods: length and charAt.

Examples:

```
findNonRepeat("calculate") returns 4
Explanation: The first non repeating character is u, thus, 4 is returned.

findNonRepeat("Algorithm") returns 0

findNonRepeat("aabbacdcd") returns -1
```

Solution:

Grading scheme:
Method signature: 2 pts
Return index or return -1: 2 pts
Outer for loop: 1.5 pts
initialization count: 1 pt
inner for loop:1.5 pt
condition in inner loop: 2 pts
updating count: 1 pt
checking for character repetition using count/flag: 2 pts
break/flag: 2 pts

```
public static int main(String s)
{
    boolean f = false;

    for(int i = 0; i < s.length(); i++)
      {
        int count = 0;

        for(int j = 0; j < s.length(); j++)
          if ( i != j && s.charAt(i) == s.charAt(j))
            count++;
        if (count == 0)
        {
          System.out.println(i);
          f = true;
          break;
        }
      }

    if (f == false)
      System.out.println(-1);
}
```

Other solution:

```
public static int findNonRepeating(String s)
    {
        int index=-1;
        for(int i=0; i<s.length();i++)
        {
            int count=0;

            for(int j=0; j<s.length(); j++)
            {
                if (s.charAt(i)==s.charAt(j))
                 count++;
            }

            if(count==1)
             { index=i;
               break;
             }
        }
        return index;
    }
```

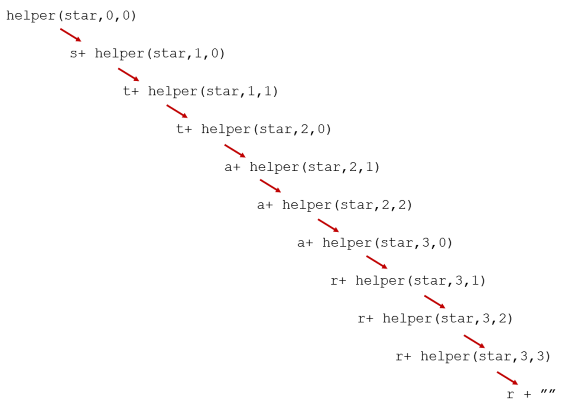
Exercise 6 (8 Marks)

Consider the following program.

```
public static String mystery(String s)
{
    return helper(s,0, 0);
}

public static String helper(String s, int index, int counter)
{
    if(s.length() == index)
        return "";

    if(index == counter)
        return s.charAt(index) + helper(s, ++index, 0);

    return s.charAt(index) + helper(s, index, ++counter);
}
```

- What is the output after executing the program for `mystery("star")`? Trace the program and show your workout.

  Solution:



  output: sttaaarrrr

- What does the program do for any input `s`? Express it with one English statement.

  Solution:

  The program generates a new string with repeated characters based on their position in the input string.

Grading scheme:
Output: 2 pts
Tracing (could be tree, or stack or any other way): 4 pts
Functionality: 2 pts

Scratch paper

Scratch paper

Scratch paper