Basic Programming

Problem 01:

There are some concepts that are considered basic programming skills, such as operators, loops and conditions. These are the building blocks of complex software. In this problem, you are asked to demonstrate your knowledge in basic programming.

```java
//problem-01/BasicProgram.java

import java.lang.*;
public class BasicProgram{
	public int sum(int a, int b){
		// your code goes here.
	}
	public double div(int a, int b){
		// your code goes here.
	}
	public void displayArray(int []array){
		// your code goes here.
	}
}
```

Requirements:

You are required to complete the aforementioned class with your code.

```
/*
* The class BasicProgram is a general class;;
*
* Signatures:
* method sum:
*	return type: integer;
*	requirements:
*		- return the sum of two numbers;
*
* method div:
*	return type: double;
*	requirements:
*		- return the div of two numbers;
*		- the number b must not be 0;
*		- If division by zero, method MUST throw an error, with the
*		message: "Not possible division by zero.".
*		- exception type: UnsupportedOperationException
```

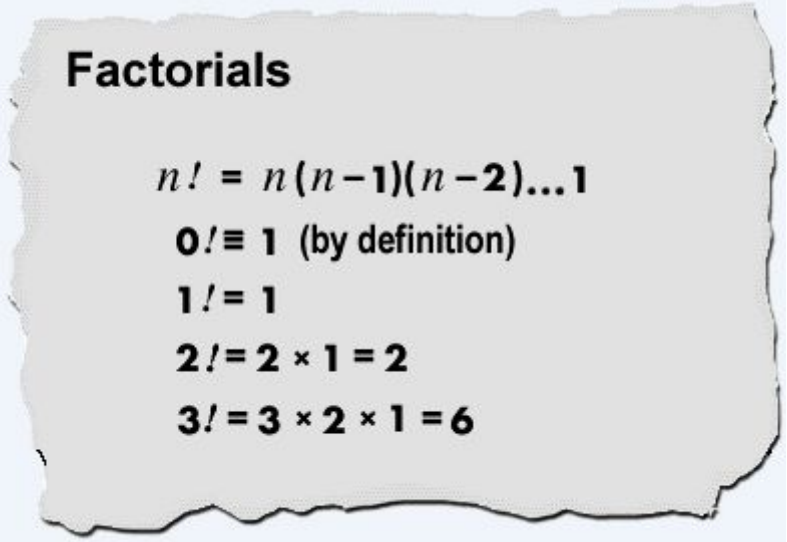
```
*           If necessary, you are allowed to check java docs, for this
*           Exception (java.lang.UnsupportedOperationException)
*
*     Note: your method just has to throw the exception, the message will
*     be handled by the testing script in Websubmission;
*
* method displayArray:
*     return type: void;
*     requirements:
*           - print the values in an array;
*           - display format: "[ 1, 2, 3, 4, 5];"
*           - the number of elements in the array must not be 0;
*           - If the number of elements is zero, method MUST
*           throw an error, with message:  "No elements."
*           -exception type: UnsupportedOperationException
```

Intermediate Programming

Problem 02:

Concepts such as recursion and complex manipulation of loops are found in intermediate programming skills. In this problem, you are asked to demonstrate your knowledge in intermediate programming.

Factorial number: For an integer n greater than or equal to 1, the factorial is the product of all integers less than or equal to n but greater than or equal to 1. The factorial value of 0 is defined as equal to 1. The factorial values for negative integers are not defined.




```java
//problem-02/Factorial.java

public class Factorial{
	public int find(int x){
		// your code goes here.
	}
}
```

Requirements:

You are required to complete the aforementioned class with your code.

Specifications for the code.

```
/*
* The class Factorial is a general class;
* Signatures:
* method find:
*     return type: integer;
*     requirements:
*           - return the factorial number of the value x;
*           - the number x must not be negative;
*           - If the value is negative, method MUST throw an exception,
*           with message: "Not defined."
*           - exception type: UnsupportedOperationException
*           If necessary, you are allowed to check java docs,
*           for this Exception.
*           - Recursive solution will be award with additional marks;
```

Object Oriented Programming

Problem 03:

Object-oriented programming (OOP) is a programming language model organized around objects rather than "actions". In this problem, you are asked to demonstrate your knowledge in OOP.

The following two classes have been given to you and do not need to be changed. Copy this code into relevant files in preparation for making a child class.

```
//problem-03/Personable.java
public interface Personable{
	public void display();
}
//Note: your file Personable.java will be replaced with the testing
//script. Therefore, don't add other code to this file.
```

```
//problem-03/PersonAbstract.java
public abstract class PersonAbstract implements Personable {
	protected int age;
	protected String name;
	protected String address;

	public abstract void display();

	public abstract void setAge(int _age);
	public abstract void setName(String _name);
	public abstract void setAddress(String _address);

	public abstract int getAge();
	public abstract String getName();
	public abstract String getAddress();
}
//Note: your file PersonAbstract.java will be replaced with the testing
//script. Therefore, don't add other code to this file.
```

Requirements:

In this problem you are asked to implement the class **Person.java**;

You are required to use the above classes with your code.

Specifications for the code.

```
/*
* class: Person.java
* The class Person consists of a class that inherits from the abstract
* class PersonAbstract.
*
* - Implement accessors & mutators for all attributes defined in
* PersonAbstract;
* - Implement 1) non-parametric constructor and set
*                  (name: ”unknown”,* age: -99, address: “None”);
*              2) parametric constructor with all parameters in the order:
*                  (age, name, address)
* - Inheritance will be marked by tutors for this (4 marks)
*
* Signatures:
* method display:
*     return type: void;
*     requirements:
*           - display information about person;
*           - display format: “name: unknown, age: -99, address: None;”
```

Algorithms & Data Structure

Problem 04:

A data structure is a data organization, management and storage format that enables efficient access and modification. Algorithms are the structure that programs use to manage and deal with this data. In this problem, you are asked to demonstrate your knowledge in Algorithms and Data Structures.

```java
//problem-04/Structurable.java

public interface Structurable{
      public int [] sort(int [] array);
      public int search(String [] array, String value);
}
```

Requirements:

In this problem you are asked to implement the class **DataStructure.java**;

```
/*
* class: DataStructure.java
* The class DataStructure consists of a class that implements the
* interface Structurable
*
* Signatures:
* method sort:
*     return type: int [];
*     requirements:
*           - Implement any sorting algorithm of your preference;
*           - sort descending an array of integer values;
*           - the number of elements in the array must not be 0;
*           - If the number of elements is zero, method MUST throw an
*           exception, with message: "No elements."
*           - exception type: UnsupportedOperationException
* method search:
*     return type: int;
*     requirements:
*           - Implement any searching algorithm of your preference;
*           - search for a value in a array of Strings;
*           - return the index of a given element
*           - If the value is not existent, return -99;
*           - If there are repeated values, return the first found;
*           - If the number of elements is zero, method MUST throw an
*            exception, with message: "No elements."
*           - exception type: UnsupportedOperationException
```