```java
public interface ICommand {

/**
 * Return the command's name.
 */
	public String getName();
	/**
	 * Perform the operation of the chosen command, if all information inserted is valid.
	 */
	public void execute(String[] split);
}
```

```java
public interface IComplexCommand {

/**
 * This method stores the commands that will be performed on the string inserted on the console
 * @param Command
 */
	void addCommand(ICommand printCmd);

	/**
	 * Return the command's name.
	 */
	String getName();

/**
 * Perform the operation of the chosen command, if all information inserted is valid.
 */
	void execute(String[] split);
}
```

```java
public class CommandContainer implements IComplexCommand, ICommand {
	private int numberCommands = 0;
	private ICommand[] commands;

	public CommandContainer(int numCommand) {
		commands = new ICommand[numCommand];
		// afetação de um array com dimensão igual ao número de comandos adicionados
	}

	/**
	 * Return the command's name.
	 */
	@Override
	public String getName() {
		return "help";
	}

/**
 * This method stores the commands that will be performed on the string inserted on the console
 * @param Command
 */
	@Override
	public void addCommand(ICommand Command)
	{
			if(numberCommands < commands.length)
			{
				commands[numberCommands]=Command;
```

```
                        numberCommands++;
                    }
        }

        /**
         * Perform the operation of the chosen command, if all information inserted is valid.
         */
        @Override
        public void execute(String[] split) {
                int count=0;
                if (split[0].equals("help"))
                                // impressão dos comandos disponíveis para serem executados
                {
                        System.out.println(" print (message)\n "
                                        + "sum (value1) (value2) \n "
                                        + "minus (value1) (value2) \n "
                                        + "exit");
                }
                else    // se for escrito qualquer um dos outros comandos executa-os
                {
                        for (int idx=0; idx < numberCommands; idx++){

            // se a primeira palavra inserida corresponder a algum dos comandos, executa-os
                                if (split[0].equals(commands[idx].getName())){
                                        commands[idx].execute(split);
                                        count++;
                                }
                        }
                        if (count==0)    // se não escreveu um comando válido
                                System.out.println("Use Help to view every command\n");
                }
        }
}
```

```
/**
 * Performs the operation value1 + value2 and displays the result on the console .
 * @author FilipaG
 */
public class SumCmd implements ICommand{
/**
 * Return the command's name.
 */
        @Override
        public String getName() {
                return "sum";
        }

/**
 * If value1 and value2 are numbers, perform the operation, perform the operation value1+value2 and
 * display the result on the console
 */
        @Override
        public void execute(String[] split) {
                                // só é feito se os valores inseridos forem numéricos
                if ((split.length==3)&&(isNumeric(split[1])&&isNumeric(split[2]))){
                        double result = (Double.parseDouble(split[1]) +
Double.parseDouble(split[2]));
                        System.out.println("Result: " + result);
```

```
                                                // porque os valores inseridos podem ser double
                }
                else
                        System.out.println("Use 'help' to view every command");
        }


        /**
         * Verify if the string is a number.
         * @param str
         * @return true if the string is a number; false otherwise;
         */
        private static boolean isNumeric(String str)
                                                        // para testar se os valores são numéricos
        {
          try
          {
            double d = Double.parseDouble(str);
          }
          catch(NumberFormatException nfe)
          {
                System.out.println("NumberFormatException");
                return false;
          }
          return true;
        }
}
```

```
/**
 * Perform the operation value1-value2 and display the result on the console.
 * @author FilipaG
 */
public class MinusCmd implements ICommand{
//minus (value1) (value2): Realiza a operação value1-value2 e mostra o resultado na
consola.

        /**
         * Return the command's name.
         */
        @Override
        public String getName() {
                return "minus";
        }

        /**
         * If value1 and value2 are numbers, perform the operation, perform the operation
value1-value2 and display the result
         * on the console
         */
        public void execute(String[] split) {
                if ((split.length==3)&&(isNumeric(split[1])&&isNumeric(split[2]))){    // só
é feito se os valores inseridos forem numéricos
                        double result = (Double.parseDouble(split[1]) -
Double.parseDouble(split[2]));
                        System.out.println("Result: " + result);
                                // porque os valores inseridos podem ser double
```

```
                }
                else
                        System.out.println("Use 'help' to view every command");
        }

        /**
         * Verify if the string is a number.
         * @param str
         * @return true if the string is a number; false otherwise;
         */
        private static boolean isNumeric(String str)    // para testar se os valores são
numéricos
        {
          try
          {
            double d = Double.parseDouble(str);
          }
          catch(NumberFormatException nfe)
          {
                System.out.println("NumberFormatException");
                return false;
          }
          return true;
        }
}
```

```
/**
 * Displays the console message passed as parameter on the console .
 * @author FilipaG
 */
public class PrintCmd implements ICommand{

        /**
         * Return the command's name.
         */
        @Override
        public String getName() {
                return "print";
        }

        @Override
        public void execute(String[] split)
        {
                System.out.print("Print ");
                for (int idx = 1; idx < split.length; idx++)
                        System.out.print(split[idx] + ((idx==split.length-1) ? "\n" : " "));

        }
}
```

```
/**
 * Terminates the application.
 * Note: System.exit(0) (Forces  the application's finish.)
 * @author FilipaG
 */
public class ExitCmd implements ICommand{

        /**
         * Return the command's name.
```

```
	 */
	@Override
	public String getName() {
		return "exit";
	}

	/**
	 * Terminates the application.
	 */
	@Override
	public void execute(String[] split) {
		System.exit(1);
	}
}


import java.util.Scanner;

public class App {

	public static void main(String[] args) {

		Scanner scanner = new Scanner(System.in);

		IComplexCommand container = new CommandContainer(5);

		ICommand printCmd = new PrintCmd();
		ICommand sumCmd = new SumCmd();
		ICommand minusCmd = new MinusCmd();
		ICommand exitCmd = new ExitCmd();

		container.addCommand(printCmd);
		container.addCommand(sumCmd);
		container.addCommand(minusCmd);
		container.addCommand(exitCmd);

		System.out.println("Use '" + container.getName() + "' to view every
command\n");
		//nome do container: Help

		String cmd;

		while(true){

			cmd = scanner.nextLine();	//inserção da linha de instruções

			System.out.println();

			container.execute(cmd.split(" "));	//executa a linha de instrução
fazendo um split sobre a mesma

			System.out.println();
		}
	}
}
```

Output:

```
Use 'help' to view every command

help
 print (message)
 sum (value1) (value2)
 minus (value1) (value2)
 exit

print olá
Print olá

sum 1.5 1.9
Result: 3.4

minus
Use 'help' to view every command

minus tudo bem
NumberFormatException
Use 'help' to view every command

minus 5 9
Result: -4.0

olá tudo bem
Use Help to view every command

exit
```