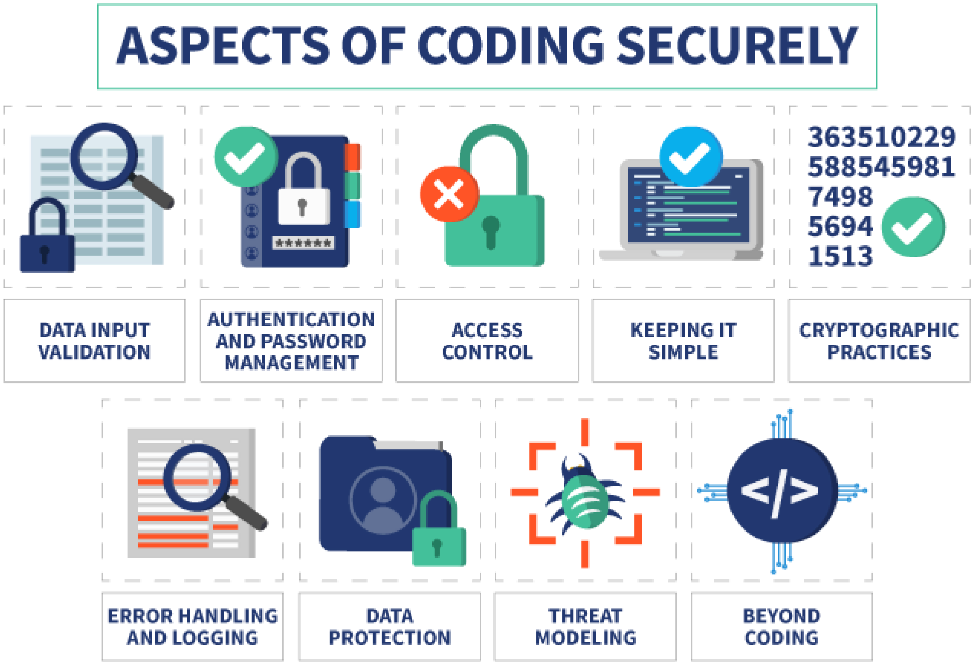
Exercises UD05



1. Security

1. Exercise 1:

   An antimalware application is both:

   1. Logical and active security.
   2. Physical and active security.
   3. Logical and passive security.

2. Exercise 2:

   List the following risks and threats according to whether they are physical or logical.

Threat or Risk	**Type (F/L)**
SPAM	
Leak in a pipe that passes over the CPD	
A nearby construction site has cut off power to the entire block	
Heavy rains cause flooding	
Due to the arrival of summer temperatures increase	
Phishing	
A person gets an identification card and gains access to the CPD without being authorized	
A person unintentionally causes a fire in the office	
A user downloads an executable and disables anti-malware software in order to run it	

3. Exercise 3:

   Indicate which of the following statements are true:

   1. It is impossible to achieve a secure, easy to use and functional system at the same time.
   2. The computer security audit does not require prior permission if we notify the problems we detect.
   3. Thanks to non-repudiation we can be sure that the information sent by the issuer is the one that has reached us.
   4. The availability of information ensures that it is accessible from anywhere and at any time.
   5. It is possible to achieve full security in a computer system.

4. Exercise 4:

   Fill in the following text with the information learned during this point.

   _____ security is in charge of managing the security of physical devices and controlling ____ elements such as temperature and relative humidity. Against security _____ deals with the security of software elements and the ____ of the user, as well as controlling malicious software, also called _____.

(a) environmental, (b) physical, (c) information, (d) logical, (e) malware

5. Exercise 5:

   Match the different vulnerabilities with the recommended protection systems:

Vulnerability	**Protection**
Malware.	Soluciones antivirus i antimalware.
floods.	Technical construction solutions.
earthquakes.	Watertight CPD.
Ransomware	Backups
Social engineering attack.	Education and training for users

6. Exercise 6:

   Indicate which of the following statements are true:

   1. A 0-day vulnerability is little known and there is not yet a patch that solves the problem.
   2. Guaranteeing 100% security is possible, you just need sufficient financial resources.
   3. The best place to place a server is one that is accessible to all company personnel at any time.
   4. Although the CPD has many security measures, it is advisable to have a second CPD or backup CPD to ensure the continuation of the business in the event of a disaster.
   5. It is better to perform system updates only when we detect problems to prevent them from destabilizing our system.

7. What is the identifier-password pair of a security system called?
   a) Profile.
   b) Credential.
   e) Authorization.
   d) Roll.

8. Which of the following physical characteristics cannot be a biometric access control?
   to the voice.
   b) The fingerprint.
   c) Facial recognition.
   d) The height.

9. Some cryptographic algorithms with the pass of time have ceased to be secure. Explain the main reasons.

10. Explain what "credential-based access control" consists of.

2. Exceptions

1. We are going to program the exception management to read 4 integer numbers by keyboard, our teacher proposes three possible solutions:

   a) A single try block containing the reading of all numbers.

   b) A try..catch block for each read

   c) Write a method to read the integer. Inside the method will be where the exception is handled.

   Explain and implement the solution that you like the most, justify in a comment why you don't like the other two solutions and what are their main drawbacks.

2. Exception handling to read two numbers of type int, a String and two numbers of type double and display the values read on the screen. One method will be performed to read an int number and another to read a double. In these methods, exception control is performed when reading from the keyboard.

3. In this case, a Java code is provided and it is requested to handle the exceptions that may be thrown.

```java
public class Excepciones {

    static Scanner sc = new Scanner(System.in);

    public static void main(String[] args) {

        double n;
        int posicion;
        String cadena ;
        double[] valores = {9.83, 4.5, -3.06, 0.06, 2.52, -11.3, 7.60, 3.00, -30.4, 105.2};

        System.out.println("Contenido del array antes de modificar:");
        for (int i = 0; i &lt; valores.length; i++) {
            System.out.printf("%.2f ", valores[i]);
        }

        System.out.print("\n\nIntroduce la posición del array a modificar: ");
        cadena = sc.nextLine();
        posicion = Integer.parseInt(cadena);

        System.out.print("\nIntroduce el nuevo valor de la posición " + posicion + ": ");
        n = sc.nextDouble();

        valores[posicion] = n;

        System.out.println("\nPosición a modificar " + posicion);
        System.out.println("Nuevo valor: " + n);
        System.out.println("Contenido del array modificado:");
        for (int i = 0; i &lt; valores.length; i++) {
           System.out.printf("%.2f ", valores[i]);
       }

   }
}
```

4. Write a program that divide two numbers received in main into `args[0]` and `args[1]`.

   Example:

   ```
   $ java dividir 10 5
   10/5 es igual a 2
   ```

   Where 10 and 5 are `args[0]` and `args[1]` respectively, that is, the parameters with which we call the divide program.

5. Justify why the following code snippet fails

   ```
   try {
      System.out.println("Introduce edad: ");
      int edad = tec.nextInt();
      if (edad &gt;= 18) {
          System.out.println("Mayor edad");
      } else {
          System.out.println("Menor edad");
      }
      System.out.println("Introduce nif");
      String nif = tec.next();
      int numero = Integer.parseInt(nif.substring(0, nif.length() - 1));
      char letra = nif.charAt(nif.length() - 1);
      System.out.println("Numero: " + numero);
      System.out.println("Letra: " + letra);
   } catch (Exception e){
       System.out.println("Debías introducir un número");
   } catch (NumberFormatException e) {
       System.out.println("El nif es incorrecto");
   }
   ```

6. Explain what will be displayed on the screen when this class is executed and why:

   ```
   public class Uno {
        private static int metodo()  {
              int valor=0;
              try  {
                    valor = valor + 1;
                    valor = valor + Integer.parseInt("42") ;
                    valor = valor + 1;
                    System.out.println("Valor al final del try: " + valor);
              } catch(NumberFormatException e)  {
                    valor = valor + Integer.parseInt ("42");
                    System.out.println("Valor al final del catch: " + valor) ;
              }
   ```

```
        finally  {
            valor = valor + 1;
            System.out.println("Valor al final de finally: " +
valor) ;
        }
        valor = valor + 1;
        System.out.println ("Valor antes del return: " + valor) ;
        return valor;
    }

    public static void main(String[] args)  {
        try {
            System.out.println (metodo());
        }  catch (Exception e)  {
            System.err.println("Excepcion en metodo()") ;
            e.printStackTrace();
        }
    }
}
```

7. Explain what will be displayed on the screen when this class is executed and why:

```
public class Dos {
    private static int metodo()  {
        int valor=0;
        try   {
            valor = valor+1;
            valor = valor + Integer.parseInt("W");
            valor = valor + 1;
            System.out.println("Valor al final del try: " +
valor);
        } catch(NumberFormatException e) {
            valor = valor + Integer.parseInt("42");
            System.out.println("Valor al final del catch: " +
valor) ;
        } finally {
            valor = valor + 1;
            System.out.println("Valor al final de finally: " +
valor) ;
        }
        valor = valor + 1;
        System.out.println ("Valor antes del return: " + valor) ;
        return valor ;
    }

    public static void main (String[] args)  {
        try  {
          System .out.println(metodo());
        }  catch (Exception e) {
            System.err.println("Excepcion en metodo() ");
            e.printStackTrace();
        }
    }
}
```

8. Explain what will be displayed on the screen when this class is executed and why:

```java
public class Tres {
     private static int metodo()  {
           int valor = 0;
           try {
                 valor = valor +1;
                 valor = valor + Integer.parseInt("W");
                 valor = valor + 1;
                 System.out.println("Valor al final del try : " + valor);
           } catch (NumberFormatException e) {
                 valor = valor + Integer.parseInt("W");
                 System.out.println("Valor al final del catch : " + valor);
           } finally {
                 valor = valor + 1;
                 System.out.println("Valor al final de finally: " + valor);
           }
           valor = valor + 1;
           System.out.println ("Valor antes del return: " + valor);
           return valor ;
     }

     public static void main (String[ ] args)
     {
           try {
                 System.out.println(metodo ());
           } catch (Exception e) {
                 System.err.println("Excepcion en metodo()") ;
                 e.printStackTrace();
           }
     }
}
```

9. Explain what will be displayed on the screen when this class is executed and why:

```java
import java.io.*;

public class Cuatro{
      private static int metodo()  {
            int valor = 0;
            try {
                  valor = valor+1;
                  valor = valor + Integer.parseInt("W");
                  valor = valor + 1;
                  System.out.println("Valor al final del try : " + valor) ;
                  throw new IOException();
            } catch (IOException e)  {
                  valor = valor + Integer.parseInt("42");
                  System.out.println("Valor al final del catch : " + valor);
            } finally {
```

```
         valor = valor + 1;
         System.out.println("Valor al final de finally: " + valor);
      }
      valor = valor + 1;
      System.out.println ("Valor antes del return: " + valor);
      return valor ;
   }

   public static void main(String[] args)  {
      try {
         System.out.println(metodo());
      } catch (Exception e) {
         System.err.println("Excepcion en metodo()");
         e.printStackTrace();
      }
   }
}
```

10. Explain what will be displayed on the screen when this class is executed:
    1. If running with java Five home
    2. If run with java Five 0
    3. If running with java Five 7

```
public class Cinco {
  public static void main(String args[])  {
   try  {
      int a = Integer.parseInt(args[0]);
      System.out.println("a = " + a);
      int b=42/a;
    String c = "hola";
      char d = c.charAt(50);
   }  catch (ArithmeticException e) {
     System.out.println("div por 0: " + e);
   }  catch (IndexOutOfBoundsException e) {
     System.out.println("Índice del String fuera de límites: " + e);
   }  finally {
      System.out.println("Ejecución de finally");
   }
  }
}
```

11. Indicate what the output of the following program will be and why

```
public class Seis {
  public static void procA()  {
    try {
       System.out.println("dentro del procA");
       throw new RuntimeException("demo");
    } finally {
       System.out.println("Finally del procA");
    }
    }

   public static void procB() {
       try  {
           System.out.println("dentro del procB");
           return; 7
       } finally {
           System.out.println("finally del procB");
       }
    }

    public static void main(String args[])  {
        try  {
            procA();
        } catch(Exception e) {
            procB();
        }
     }
}
```

12. Indicate what the output of the following program will be and why

```
public class Siete {
   public static void metodo() {
       try  {
           throw new NullPointerException("demo");
       } catch (NullPointerException e) {
           System.out.println("capturada en método");
           throw e;
       }
    }

    public static void main (String args[])  {
        try   {
            metodo();
        }  catch(NullPointerException e)  {
           System.out.println("capturada en main " + e);
        }
    }
}
```

13. Write a program that plays with the user to guess a number. The computer must generate a number between 1 and 500, and the user has to try to guess it. To do this, each time the user enters a value, the computer must tell the user if the number to be guessed is greater or less than the number entered by the user. When it manages to guess it, it must indicate it and print on the screen the number of times the user has tried to guess the number. If the user enters something that is not a number, they must indicate it on the screen, and count it as an attempt. It must be checked that the reading that is performed is really an integer, and if not, give an error message and request it again. *Scanner* indicates that it failed to recognize the input by throwing the exception `InputMismatchException`.

14. Try to guess the screen output that the following program would produce, explain why happens this:

```java
public class EjemploExcepciones {

    public static int devuelveNumero(int num) {
        try {
             if (num % 2 == 0) {
               throw new Exception("Lanzando excepcion");
            }
            return 1;
        } catch (Exception ex) {
            return 2;
        } finally {
            return 3;
        }
       }

    public static void main(String[] args) {
        System.out.println(devuelveNumero(1));
    }
}
```

3. Log4j2

1. Write a simple program using Log4j2 that prints ALL error levels to the console.
2. Using the program from the exercise 1, generate a manual configuration file to print FATAL errors to SYSTEM_ERR and the rest of the error levels to SYSTEM_OUT. (Add comments in the XML with `<!--Your comment-->`)
3. Using the program from the exercise 1, generate a manual configuration file in which FATAL errors go to a text file, ERROR level errors appear red on the console (system_err) and the rest appear on the normal console (system_out).
4. Using the previous file (exercise 3) change the Output Pattern so that the first thing that appears in the LOG line are your initials.

4. Information sources

- Wikipedia
- Programación de servicios y procesos - FERNANDO PANIAGUA MARTÍN [Paraninfo]
- Programación de Servicios y Procesos - ALBERTO SÁNCHEZ CAMPOS [Ra-ma]
- Programación de Servicios y Procesos - Mª JESÚS RAMOS MARTÍN - [Garceta] (1ª y 2ª Edición)
- Programación de servicios y procesos - CARLOS ALBERTO CORTIJO BON [Sintesis]
- Programació de serveis i processos - JOAR ARNEDO MORENO, JOSEP CAÑELLAS BORNAS i JOSÉ ANTONIO LEO MEGÍAS [IOC]
- GitHub repositories:
    - https://github.com/ajcpro/psp
    - https://oscarmaestre.github.io/servicios/index.html
    - https://github.com/juanro49/DAM/tree/master/DAM2/PSP
    - https://github.com/pablohs1986/dam_psp2021
    - https://github.com/Perju/DAM
    - https://github.com/eldiegoch/DAM
    - https://github.com/eldiegoch/2dam-psp-public
    - https://github.com/franlu/DAM-PSP
    - https://github.com/ProgProcesosYServicios
    - https://github.com/joseluisgs
    - https://github.com/oscarnovillo/dam2_2122
    - https://github.com/PacoPortillo/DAM_PSP_Tarea02_La-Cena-de-los-Filosofos