Universidad
Carlos III de Madrid

SeTI Group · *Computer Systems Dept.*

Universidad Carlos III de Madrid

Hash functions Annex

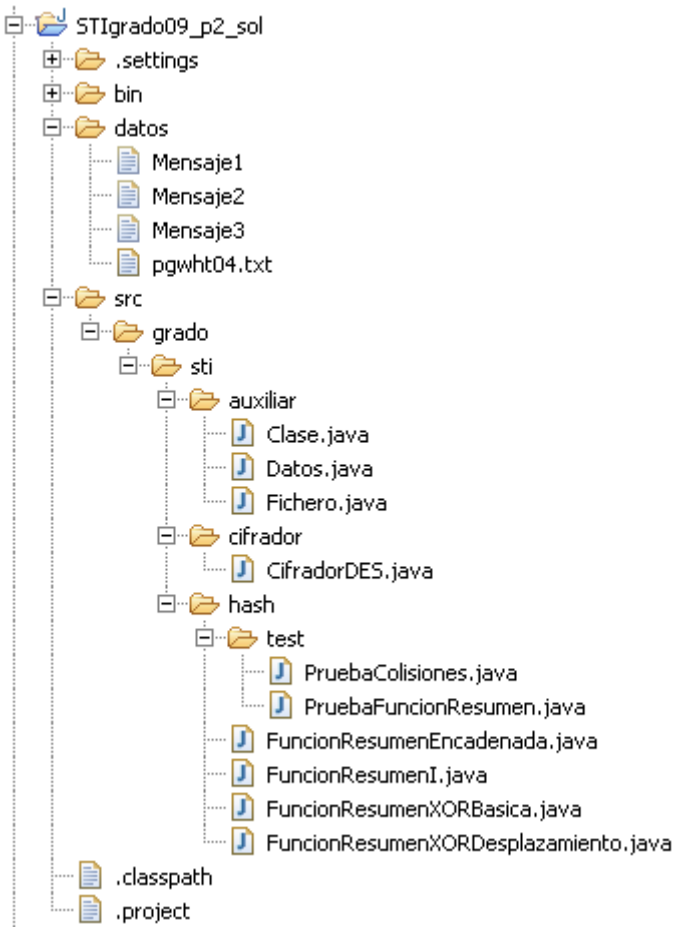
IT Security practices
Course 2010/2011

Jorge López Hernández-Ardieta / Adapted by Iván Rivera 2010-2011

1. Table of contents

2. Java Project Structure

Apart from this document, a Java Project with a set of class a resources to be used by the student when developing the practice is given in AulaGlobal2. The project, **STIgrado10_p3**, hash the following folder structure:



Resource	Description	Comments
`datos`	Directory containing the application input/output resources (files).	All the input and output files must be stored in this folder.

`grado.sti.auxiliar`	Package containing some auxiliary classes.	The student must uniquely implement some methods from the class **`Datos.java`**
`grado.sti.cifrador`	Basic implementation of a DES cipher.	The class **`CifradorDES.java`** mustn't be modified by the student.
`grado.sti.hash`	Package containing all the hash functions that must be developed in this practice.	The student must implement the specified methods from the classes: **`FuncionResumenXORBasica.java`**, **`FuncionResumenXORDesplazamiento.java`** **`FuncionResumenEncadenada.java`** The student mustn't modify the interface **`FuncionResumenI.java`**.
`grado.sti.hash.test`	Package containing the execution test classes.	The student must implement the class **`PruebaColisiones.java`**. The **`PruebaFuncionResumen.java`** mustn't be modified

Along with the project, the javadoc (Sun standard generated documentation) is done, so the student can easily access to each class and method description.

2.1 Java program exectution

In order to know the parameters and the format asking in the test classes, exectute:

```
> Java PruebaFuncionResumen
```

```
> Java PruebaColisiones
```

3. Simplifications

- Message to be processed must be as much 255 bytes length. It's recommended to create messages containing only ASCII characters ('a'-'z', 'A'-'Z', 0-9), so that each of them take 1 just byte.

- The processed message must be decomposed into block segment of a predefined length. This length, for this practice, will always be 8 bytes.

- The hash value length must also be 8 bytes.

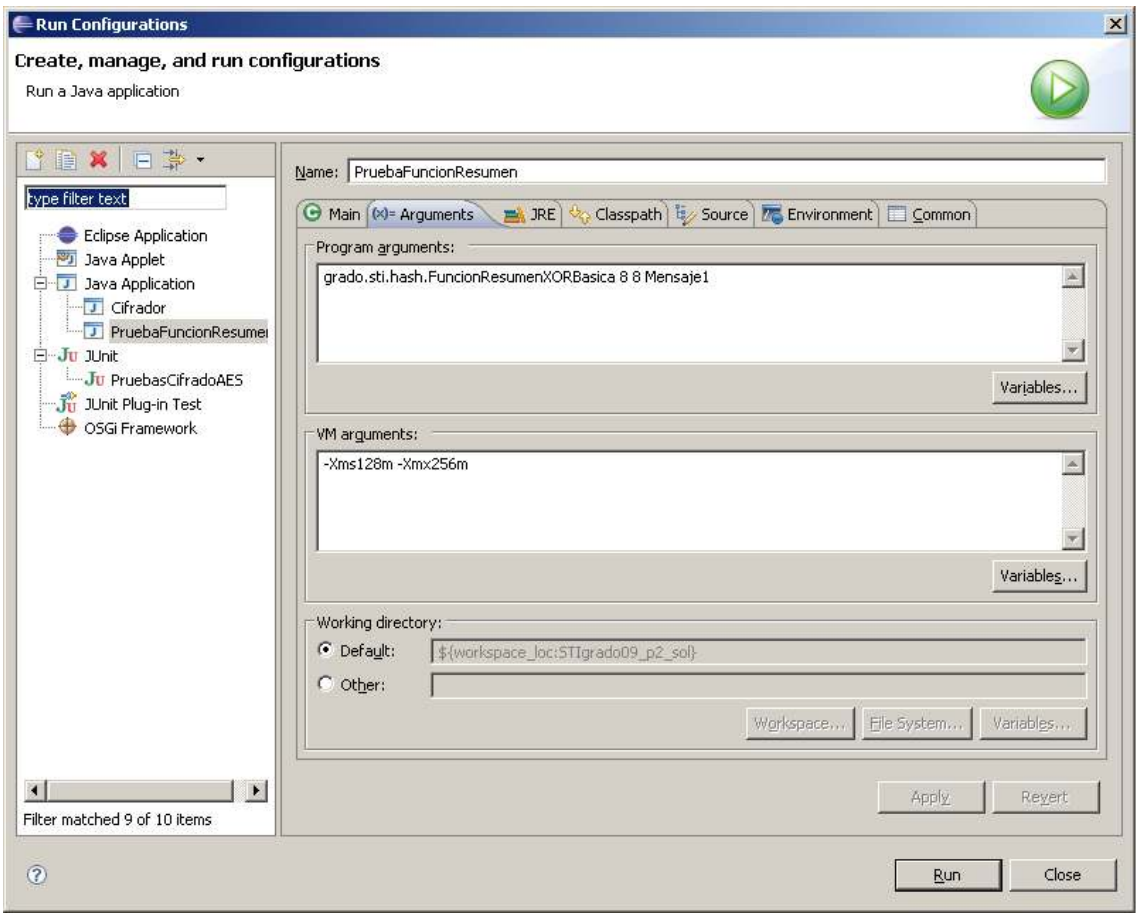
4. Advices

- To assure that the desired bits take the 0 value in the shift operations, we recommend the use of masks (0xFE, 0x80, 0x01, etc.) and binary shift operations (<<, >>, >>>).

- In order to know if the shift operations are properly working, it's recommended to visualize the bytes in the hexadecimal base. For that purpose, some given methods in the class **`Datos.java`** can be used or show by console the resultant values obtaining using the next String format of Java:

```
System.out.printf("%x", miByte);
```

- Grow up the Java Virtual Machine size when calculating collisions. To do that, you must include in the parameters configuration the minimum memory size (-Xms) and the maximum memory size (-Xmx).



- When calculating the collisions, it's recommend to use the Java class **`HashMap.java`**, being the keys the hash values and the values the number of messages found for this hash value.

- It's necessary to include the folder `datos` into the classpath of the project, using it's properties (*Java Build Path* → *Source* → *Add* → select folder *datos*).

5. Hash function results

To verify the correct implementation of the hash functions, we give some hash values that must be computed by each function when hashing the message: `HOLAQUETALESTAS?`

Hash function	Generated hash value
`FuncionResumenXORBasica`	`09:03:09:12:05:14:16:6B`
`FuncionResumenXORDesplazamiento`	`D1:D2:DD:D1:F6:EB:D9:97`
`FuncionResumenEncadenada`	`FB:A6:A4:07:1B:0A:7E:98`