UNIVERSITÀ DEGLI STUDI DI FIRENZE
Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea Magistrale in Informatica

Corso di Tecniche Avanzate di Programmazione

CIPHERS: IMPLEMENTAZIONE DI ALCUNI CIFRARI STORICI

MARCO BURACCHI

Prof. Lorenzo Bettini

Anno Accademico 2016-2017

INDICE

1

CENNI PRELIMINARI

INTRODUZIONE

In questo progetto sono stati implementati cinque storici cifrari a chiave simmetrica. I cifrari implementati sono:

- Shift cipher (altrimenti noto come cifrario "Giulio Cesare")
- Vigenere cipher (l'evoluzione del cifrario di Giulio Cesare)
- Substitution cipher
- Affine cipher
- OneTimePad cipher (il cifrario perfetto)

STRUTTURA PROGETTO

Il progetto consta di otto classi. Cinque rappresentano i cinque cifrari, una contiene solamente delle costanti utilizzate più o meno da tutte e cinque le classi principali, una è la classe che implementa il parser che si occupa di gestire i messaggi e le varie stringhe utilizzate dal programma e infine un'interfaccia per le possibili implementazioni di parser differenti.

Tutte le classi sono state create con la metodologia test-driven e testate attraverso unit-testing.

2

TESTING

UNIT TESTING

Andiamo a vedere nel dettaglio i test con i quali sono state implementate le varie classi.

Shift cipher

Lo shift cipher è il cifrario più semplice. L'unica cosa che fa è spostare la lettere da cifrare di un certo numero di posizioni avanti nell'alfabeto. Tale numero di posizioni è la chiave del cifrario. Per decodificare il messaggio basterà semplicemente tornare indietro del numero di posizioni indicato dalla chiave. Se per esempio il messaggio è "ciao" e la chiave è 3, il messaggio cifrato sarà "fldr".

Considerando di ottenere solamente stringhe di caratteri cifrabili dal parser (nell'ultima sezione di questo capitolo verrà spiegato il suo utilizzo tramite mocking), le uniche due funzionalità richieste a questo cifrario sono quelle di cifratura e decifratura. Tali funzionalità sono state testate nel caso di uno spostamento di zero posizioni (non ottenendo una cifratura), di una posizione, di ventisei posizioni, di ventisette posizioni e di un numero negativo di posizioni(-1). Come caso limite è stata testata anche la cifratura della stringa vuota (il numero di posizioni è ininfluente dato che la cifratura terminerà immediatamente).

Testando anche la decodifica di tutti questi casi sono state coperte le possibili modalità di cifratura/decifratura di una stringa.

Mocking

MUTATION TESTING

INTEGRATION TESTING

TEST SULL'INTERFACCIA

3

PLUG-IN UTILIZZATI

MAVEN

TRAVIS

COVERALLS

SONARQUBE

4

CONCLUSIONI

A

SVOLGIMENTO COMPLETO ESERCIZIO 5.4

$$
\begin{aligned}
\mathbf{SSSSSSS} &= (\lambda xyz.xz(yz))\mathbf{SSSSSS} \\
&= (\lambda yz.\mathbf{S}z(yz))\mathbf{SSSSS} \\
&= (\lambda z.\mathbf{S}z(\mathbf{S}z))\mathbf{SSSS} \\
&= \mathbf{SS}(\mathbf{SS})\mathbf{SSS} \\
&= (\lambda yz.\mathbf{S}z(yz))(\mathbf{SS})\mathbf{SSS} \\
&= (\lambda z.\mathbf{S}z(\mathbf{SS})z)\mathbf{SSS} \\
&= \mathbf{SS}(\mathbf{SSS})\mathbf{SS} \\
&= (\lambda yz.\mathbf{S}z(yz))(\mathbf{SSS})\mathbf{SS} \\
&= (\lambda z.\mathbf{S}z(\mathbf{SSS}z))\mathbf{SS} \\
&= \mathbf{SS}(\mathbf{SSSS})\mathbf{S} \\
&= (\lambda yz.\mathbf{S}z(yz))(\mathbf{SSSS})\mathbf{S} \\
&= (\lambda z.\mathbf{S}z(\mathbf{SSSS}z))\mathbf{S} \\
&= \mathbf{SS}(\mathbf{SSSSS}) \\
&= (\lambda yz.\mathbf{S}z(yz))(\mathbf{SSSSS}) \\
&= \lambda z.\mathbf{S}z(\mathbf{SSSSS}z) \\
&= \lambda z.(\lambda ya.a(ya))(\mathbf{SSSSS}z) \\
&= \lambda za.aa(\mathbf{SSSSS}aa) \\
&= \lambda za.aa((\lambda yb.\mathbf{S}b(yb))\mathbf{SSS}zz) \\
&= \lambda za.aa((\lambda b.\mathbf{S}b(\mathbf{S}b))\mathbf{SS}zz) \\
&= \lambda za.aa(\mathbf{SS}(\mathbf{SS})\mathbf{S}aa) \\
&= \lambda za.aa((\lambda yb.\mathbf{S}b(yb))\mathbf{SSS}zz) \\
&= \lambda za.aa((\lambda b.\mathbf{S}b(\mathbf{S}b))\mathbf{SS}zz) \\
&= \lambda za.aa(\mathbf{SS}(\mathbf{SS})\mathbf{S}aa)
\end{aligned}
$$

$$= \lambda z a.aa((\lambda y b.\mathbf{S}b(yb))(\mathbf{SS})\mathbf{S}zz)$$
$$= \lambda z a.aa((\lambda b.\mathbf{S}b(\mathbf{SS}b))\mathbf{S}zz)$$
$$= \lambda z a.aa(\mathbf{SS}(\mathbf{SSS})aa)$$
$$= \lambda z a.aa((\lambda y b.\mathbf{S}b(yb))(\mathbf{SSS})zz)$$
$$= \lambda z a.aa((\lambda b.\mathbf{S}b(\mathbf{SSS}b))zz)$$
$$= \lambda z a.aa(\mathbf{S}a(\mathbf{SSS}a)a)$$
$$= \lambda z a.aa((\lambda y b.bb(yb))(\mathbf{SSS}z)$$
$$= \lambda z a.aa((\lambda b.bb(\mathbf{SSS}bb))z)$$
$$= \lambda z a.aa(aa(\mathbf{SSS}a))$$
$$= \lambda z a.aa(aa((\lambda y b.\mathbf{S}b(yb))\mathbf{S}zz))$$
$$= \lambda z a.aa(aa(\mathbf{S}a(\mathbf{S}a)a))$$
$$= \lambda z a.aa(a((\lambda y b.bb(yb))(\mathbf{S}z)z))$$
$$= \lambda z a.aa(aa((\lambda b.bb(\mathbf{S}bb))z))$$
$$= \lambda z a.aa(aa(aa(\mathbf{S}aa)))$$
$$= \lambda z a.aa(aa(aa((\lambda y b.bb(yb))z)))$$
$$= \lambda z a.aa(aa(aa(\lambda b.bb(bb))))$$