1506
UNIVERSITÀ
DEGLI STUDI
DI URBINO
CARLO BO

Corso di Laurea in Informatica Applicata

Programmazione Logica e Funzionale - Progetto sessione invernale a.a. 2021/22

Sviluppatori:

Simone Cossi, matricola: 290796

Laura Franchi, matricola: 293476

1 Specifica del Problema

Scrivere in linguaggio Haskell e in linguaggio Prolog un programma che implementa il cifrario di Vigenère e in particolare che permette, sulla base di una scelta effettuata dall'utente, di:

- Cifrare una parola o un testo tramite una chiave, che può anch'essa essere una parola o un testo, entrambi inseriti dall'utente
- Decifrare una parola o un testo tramite una chiave che può essere una parola o un testo, entrambi inseriti dall'utente

2 Analisi del Problema

2.1 Dati di ingresso del problema

Per entrambe le funzionalità, i dati di ingresso del problema sono rappresentati da:

- Una funzionalità scelta dall'utente tra le due previste nella specifica,
- una sequenza di caratteri (composti da lettere maiuscole, minuscole e spazi) rappresentante la chiave di codifica o decodifica,
- una sequenza di caratteri (composti da lettere maiuscole, minuscole e spazi) rappresentante il testo in chiaro o criptato.

2.2 Dati di uscita del problema

L'unico dato di uscita del problema, per entrambe le funzionalità, è rappresentato da una sequenza di caratteri (composti da lettere e spazi) che rappresenta:

- il testo cifrato per quanto riguarda la prima funzionalità,
- il testo decifrato per quanto riguarda la seconda funzionalità.

2.3 Relazioni intercorrenti tra i dati del problema

Vi è una corrispondenza tra le lettere dell'alfabeto e i numeri naturali:

'a' → 0, 'b' → 1 , 'c' → 3 , … , 'z' → 25

L'operazione di cifratura e decifratura secondo il cifrario di Vigenère, si basa sulla suddetta corrispondenza, in particolare:

1. si sommano i numeri corrispondenti alle lettere del testo e della chiave, indichiamo per comodità il risultato come ***x***
2. si effettua poi l'operazione ***x mod 26***

Così facendo si ottengono i caratteri cifrati.

Il procedimento della decifratura è sostanzialmente lo stesso, con la sola differenza che nel punto 1 si effettua la sottrazione invece che la somma.

L'operazione ***x mod 26*** effettuata al punto 2 restituisce come risultato il resto della divisione euclidea tra x e 26. Questo passaggio è necessario perché:

- Se **0 < *x* < *26*** allora ***x mod 26 = x*.** Ovvero, x sarà esattamente l'indice del nuovo carattere della stringa,
- Se ***x* > *26* oppure *x* < *0*** allora ***x mod 26*** produrrà l'indice del nostro nuovo carattere. Ovvero se siamo fuori dagli indici dell'alfabeto (che vanno da 0 a 25 come detto sopra), l'operazione modulo farà rientrare x all'interno di questi ultimi.

NB. Gli spazi:

- Nella chiave non criptano/decriptano
- Nel testo da cifrare/decifrare rimangono tali

3 Progettazione dell'Algoritmo

3.1 Scelte di Progetto

Le sequenze di caratteri rappresentanti il testo, la chiave di codifica e il testo cifrato o decifrato sono rappresentati mediante l'uso di liste di caratteri. Queste liste sono allocate dinamicamente in quanto non si conosce a priori la loro lunghezza. La scelta della funzionalità da far eseguire al programma è rappresentata, invece, da un numero intero.

3.2 Passi dell'Algoritmo

I passi dell'algoritmo per risolvere il problema sono i seguenti:

- Acquisire la funzionalità che l'utente vuole eseguire
- In base alla scelta effettuata dall'utente:
  - Cifrare:
    - Acquisire la chiave di codifica
    - Acquisire il testo in chiaro
    - Calcolare l'indice della lettera del testo (da 0 a 25)
    - Calcolare l'indice della lettera della chiave (da 0 a 25)
    - Sommare i due indici, e applicare mod26
    - Calcolare la lettera corrispondente al risultato ottenuto al passo precedente
  - Decifrare:
    - Acquisire la chiave di decodifica
    - Acquisire il testo cifrato
    - Calcolare l'indice della lettera del testo (da 0 a 25)
    - Calcolare l'indice della lettera della chiave (da 0 a 25)
    - Sottrarre i due indici, e applicare mod26
    - Calcolare la lettera corrispondente al risultato ottenuto al passo precedente

4 Implementazione dell'Algoritmo

File sorgente vigenere.hs:

```
{- Programma Haskell che implementa il cifrario di Vigenère -}

               {- importazione delle librerie -}

{- Libreria necessaria per utilizzare:
       - toLower che converte una lettera nella corrispondente
         lettera minuscola -}
import Data.Char

{- Libreria necessaria per utilizzare i valori di tipo Maybe.
   Un valore Maybe a, può contenere il valore di a (rappresentato come Just a)
   oppure può essere vuoto (rappresentato come Nothing).
   Abbiamo utilizzato la seguente funzione di questa libreria:
       - fromMaybe che accetta un predefinito e un valore Maybe.
         Se il valore Maybe è Nothing, restituisce il valore predefinito;
         in caso contrario, restituisce il valore contenuto nel Maybe. -}
import Data.Maybe

{- Libreria necessaria per utilizzare:
       -  elem che verifica se un elemento si trova nella lista
       -  cycle che permette di rendere circolare una lista finita;
       -  lookup che cerca una chiave in un elenco di associazione e
          restituisce l'associazione a quella chiave;
       -  elemIndex che restituisce l'indice del primo elemento nella lista data,
          oppure restituisce Nothing se l'elemento non compare nella lista;
       -  zipWith che restituisce una lista formata dall'unione di due liste
     tramite la funzione data come primo argomento. -}
import Data.List

{- Libreria necessaria per utilizzare:
       -  readMaybe che ha successo solo se restituisce esattamente un valore. -}
import Text.Read


               {-   MAIN   -}

{- Programma che chiede all'utente di scegliere se cifrare o decifrare un testo
   in base ad una chiave.
   Fatta la scelta il programma prosegue chiedendo all'utente di inserire la
   chiave e il testo da elaborare.
   I dati vengono validati e in caso non siano accettabili viene chiesto
   all'utente di reinserirli.
   Conclude comunicando all'utente il testo criptato o decriptato secondo
   la chiave inserita -}

main :: IO()
main = do
        putStrLn "\nQuesto programma Haskell contiene al suo interno il cifrario di Vigenère"
        putStrLn "Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave."
        putStrLn "\nVengono accettate sia lettere maiuscole che minuscole, "
        putStrLn "tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo."
        putStrLn "\nScrivere:  1       per cifrare una parola o un messaggio"
        putStrLn "Scrivere:  2       per decifrare una parola o un messaggio"
        putStrLn "Scrivere un qualsiasi altro numero per terminare il programma\n"

{-    if utilizzato per far scegliere all'utente se cifrare o decifrare un testo    -}
        opzione <- acquisisciInt
        if opzione == 1
            then do
                    putStrLn "È stato scelto di cifrare un messaggio\n"
```

```
                    putStrLn "Scrivere la chiave tra \"\""
                    chiave <- acquisisciStringa
                    putStrLn "\nScrivere il testo da cifrare tra \"\""
                    testo <- acquisisciStringa
                    putStrLn "\nIl testo cifrato è:"
                    putStrLn (vigenere chiave testo)
                    putStrLn "\n"
          else if opzione == 2
             then do
                      putStrLn "È stato scelto di decifrare un messaggio\n"
                      putStrLn "Scrivere la chiave tra \"\""
                      chiave <- acquisisciStringa
                      putStrLn "\nScrivere il testo da decifrare tra \"\""
                      testo <- acquisisciStringa
                      putStrLn "\nIl testo decifrato è:"
                      putStrLn (unvigenere chiave testo)
                      putStrLn "\n"
          else putStrLn "\nProgramma terminato\nFate ripartire il programma se desiderate continuare\n\n"


               {- VALIDAZIONE INPUT -}

{- Funzione che, acquisiti dei caratteri da tastiera, verifica se sono solamente numeri interi
   Nel caso dovessero essere stati inseriti altri valori viene comunicato all'utente e
   si attende per nuovi dati in input -}

acquisisciInt :: IO Int
acquisisciInt = do
  testo <- getLine
  case readMaybe testo of
    Just x -> return x
    Nothing -> putStrLn "\nErrore\nNb. Inserire solo numeri interi:\n">>acquisisciInt


{- Funzione che, acquisiti dei caratteri da tastiera, verifica se sono state inserite
   solo lettere MAIUSCOLE, minuscole e spazi
   Nel caso fossero stati inseriti altri caratteri al di fuori di questi ultimi verrà chiesto all'utente
   di inserire un nuovo testo -}

acquisisciStringa :: IO String
acquisisciStringa = do
  inputTesto <- getLine
  let testo = read inputTesto
  case controlloTesto testo of
   True -> return testo
   False -> putStrLn "\nErrore\nInserire solo lettere MAIUSCOLE, minuscole e spazi:\n">>acquisisciStringa


{- Funzione che tramite un'altra funzione controlla che tutti i caratteri
   di una lista rientrino tra le possibilita' accettate -}

controlloTesto :: [Char] -> Bool
controlloTesto = all controlloLettera


{- Funzione che controlla se un carattere rientra tra le lettere MAIUSCOLE, minuscole o spazio
   se il carattere rientra tra questi la funzione ritorna True, altrimenti False  -}

controlloLettera :: Char -> Bool
controlloLettera x
     | x `elem` [' '] = True
     | x `elem` ['a'..'z'] = True
     | x `elem` ['A'..'Z'] = True
     | otherwise           = False
```

```
          {- CIFRATURA E DECIFRATURA -}

{- Restituisce il valore della posizione di una lettera nell'alfabeto
   Parte da 0 perciò 'a' avrà valore 0
   Possono essere lette sia lettere minuscole che maiuscole  -}

posizione :: Char -> Maybe Int
posizione carattere = elemIndex (toLower carattere) ['a'..'z']


{- Trova l'n-esimo carattere dell'alfabeto e lo racchiude tra indici
   Restituisce solamente lettere minuscole -}

lettera :: Int -> Maybe Char
lettera indice = lookup (abs $ indice `mod` 26) (zip [0..] ['a'..'z'])


{- Data una chiave, crittografa la stringa data utilizzando la crittografia di Vigenère
   Una chiave più corta rispetto al testo da cifrare verrà ripetuta in modo da poter
   cifrare tutto il testo
   Restituirà sempre lettere minuscole poiché utilizza 'lettera'  -}

vigenere :: String -> String -> String
vigenere = zipWith shift.cycle
    where
      shift chiave testo =
          fromMaybe testo $
          do posT <- posizione testo
             posC <- posizione chiave
             lettera $ posT + posC


{- Questa funzione è essenzialmente l'inverso della precedente 'vigenere'  -}

unvigenere :: String -> String -> String
unvigenere = zipWith unshift.cycle
    where
      unshift chiave testo =
          fromMaybe testo $
          do posT <- posizione testo
             posC <- posizione chiave
             lettera $ posT - posC
```

File sorgente vigenere.pl

```
/* Programma Prolog che implementa il cifrario di Vigenère */

/*      MAIN     */

/* funzione main che chiede all'utente se vuole cifrare o decifrare un testo
   in base alla scelta dell'utente chiama la funzione per cifrare o decifrare un messaggio */

main :-
       nl,
       write('Questo programma Prolog contiene al suo interno il cifrario di Vigenere'), nl,
       write('Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:'), nl,
       write('Scrivere 1 per cifrare una parola o un messaggio'), nl,
       write('Scrivere 2 per decifrare una parola o un messaggio'), nl,
       validazione_scelta(Scelta),
       if_scelta(Scelta, cifrare, decifrare).


/*      FUNZIONI DI SUPPORTO   */

/* funzione che in base al valore di 'C' esegue 'I1' o 'I2' */

if_scelta(C, I1, _) :- C=1, !, I1.
if_scelta(C, _, I2) :- C=2, !, I2.


/* funzione che in base al valore di 'Condizione' esegue 'I1' o 'I2'
   I numeri utilizzati nelle operazioni corrispondono a dei caratteri in codice ASCII  */

if_Mm(Condizione, I1, _) :-  (Condizione < 91, Condizione > 64), !, I1.
if_Mm(Condizione, _, I2) :-  (Condizione > 96, Condizione < 123), !, I2.


/* funzione che in base al valore di 'Condizione' esegue 'I1' o 'I2'
   Il numero utilizzato nelle operazioni corrisponde a allo spazio in codice ASCII  */

if_spazio(Condizione, I1, _) :- (Condizione \= 32), !, I1.
if_spazio(Condizione, _, I2) :- (Condizione == 32), !, I2.


/* funzione che in base al valore di 'c' esegue 'I1' o 'I2' */
if_validazione(C, I1, _) :- C=1, !, I1.
if_validazione(C, _, I2) :- C=0, !, I2.


/* funzione che controlla la lunghezza della chiave e del testo
       - chiave > testo -> concateno la chiave a sé stessa in modo da ripeterla
       - chiave < testo -> taglio la chiave in base alla lunghezza del testo
       - chiave = testo -> restituisco la chiave in modo che abbia la stessa lunghezza del testo */
if_lunghezza(Testo, Chiave, R) :-
       (length(Chiave, C), length(Testo, T), C < T),
       (append(Chiave,Chiave, Risultato),
       if_lunghezza(Testo, Risultato, R)),
       !.

if_lunghezza(Testo, Chiave, R) :-
       (length(Chiave, C), length(Testo, T), C > T),
       (length(Testo, Lrisultato),
       dividi(Lrisultato, Chiave, X, _),
       if_lunghezza(Testo, X, R)),
       !.
```

```
if_lunghezza(Testo, Chiave, R) :-
	(length(Chiave, C), length(Testo, T), C == T),
	R = Chiave,
	!.

/* funzione che sottrae a 'D' 65 o 97 in modo che quando è utilizzata
   per la cifratura o decifratura possa scorrere correttamente l'alfabeto
   I numeri utilizzati nelle operazioni corrispondono a dei caratteri in codice ASCII  */

maiuscola(D, D1) :- D1 is D-65.
minuscola(D, D1) :- D1 is D-97.


/* funzione che divide una stringa in due parti nella posizione passatole tramite 'Indice' */

dividi(Indice, Lista, Sinistra, Destra) :-
   length(Sinistra, Indice),
   append(Sinistra, Destra, Lista).


/*      VALIDAZIONE    */

/* Funzione che acquisisce dei caratteri e controlla se è stato inserito
   solamente '1' o '2'
   nel caso l'utente abbia inserito altro gli viene comunicato di reinserire '1' o '2'*/
validazione_scelta(Input) :-
	repeat,
		read(Input), nl,
	(
		Input == 1 , !
	;	Input == 2 , !
	;	write('input non valido, inserire solamente 1 o 2:'), nl, fail
	).


/* funzione che acquisisce una stringa e controlla che sia composta solo da
   lettere non accentate e spazi
   nel caso contenga altro chiede all'utente di inserire una nuova stringa */

validazione_input(Input) :-
	repeat,
		read(Input),
	(   validazione_lista(Input) -> true , !
	;   write('\ninput non valido, inserire solamente lettere maiuscole, minuscole e spazi:'), nl,
fail
	).

/* funzione ricorsiva in modo che 'validazione_lettera' possa scorrere tutte la lista */

validazione_lista([]).
validazione_lista([TESTA|CODA]) :-
	validazione_lettera(TESTA),
	validazione_lista(CODA).

/* funzione che controlla ogni singolo carattere,
   se il carattere non rientra tra quelli ammessi avvisa l'utente di dover inserire un nuovo input
   I numeri utilizzati nelle operazioni corrispondono a dei caratteri in codice ASCII  */

validazione_lettera(32).
validazione_lettera(Lettera):- between(65, 90, Lettera).
validazione_lettera(Lettera):- between(97, 122, Lettera).
```

```
/*            CIFRATURA       */

/* funzione che chiede all'utente di inserire la chiave di cifratura e il testo da cifrare
   i due input vengono validati e quando entrambi vengono ritenuti accettabili
   si chiama la funzione per cifrare il messaggio */

cifrare :-
       write('Hai scelto di cifrare un messaggio'), nl,
       write('NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""'), nl,
       write('esempio: "Chiave o Testo"'), nl,
       write('Inserisci la chiave di codifica:'), nl,
       validazione_input(Chiave), nl,
       write('Inserisci il messaggio che vuoi cifrare:'), nl,
       validazione_input(Messaggio),
       if_lunghezza(Messaggio, Chiave, NuovaChiave),
       vigenere(Messaggio, NuovaChiave, Cifrato),
       write('\nIl messaggio cifrato e\':\t'), write(Cifrato).


/* funzione che cripta un messaggio tramite una chiave*/

vigenere(Testo, Chiave, Cifrato) :-
       nonvar(Testo),
       nonvar(Chiave),
       vigenere_lista(Testo, Chiave, L1),
       name(Cifrato, L1).


/* funzione che controlla se la lettera che sta andando a criptare è uno spazio:
       se fosse uno spazio non verrebbe cifrata e si procede con quelle successive
       se fosse una lettera si procede con la cifratura e poi si procede con le lettere successive
       funziona tramite ricorsione */

vigenere_lista([], [], []).
vigenere_lista([TESTA|CODA], [TESTA1|CODA1], [TESTA2|CODA2]) :-
       if_spazio(TESTA,
       vigenere_lista_lettera([TESTA|CODA], [TESTA1|CODA1], [TESTA2|CODA2]),
       vigenere_lista_spazio([TESTA|CODA], CODA1, [TESTA2|CODA2])).


/* funzione che viene utilizzata quando una lettera risulta essere uno spazio
       (sia del testo che della chiave)
   restituisce la lettera originale o uno spazio in base al suo utilizzo e procede
   con la cifratura delle lettere successive */

vigenere_lista_spazio([], _, []).
vigenere_lista_spazio([TESTA|CODA], CODA1, [TESTA2|CODA2]) :-
       TESTA2 is TESTA,
       vigenere_lista(CODA,CODA1, CODA2).


/* funzione che viene utilizzata quando la lettera in cifratura non è uno spazio
   si verifica se la lettera della chiave è uno spazio
   - in caso fosse uno spazio si chiama la funzione vigenere_lista_spazio
   - in caso non fosse uno spazio si chiama vigenere_lettera per la cifratura della lettera
     e vigenere_lista per la cifratura delle lettere successive */

vigenere_lista_lettera([], [], []).
vigenere_lista_lettera([TESTA|CODA], [TESTA1|CODA1], [TESTA2|CODA2]) :-
       if_spazio(TESTA1,
       (if_Mm(TESTA1, maiuscola(TESTA1, TESTA3), minuscola(TESTA1, TESTA3)),
       vigenere_lettera(TESTA, TESTA3, TESTA2),
       vigenere_lista(CODA,CODA1, CODA2)),
       vigenere_lista_spazio([TESTA|CODA], CODA1, [TESTA2|CODA2])).
```

```
/* funzione che cifra una lettera in base a una chiave
   una lettera maiuscola rimarrà maiuscola mentre una minuscola rimarrà minuscola
   nonostante la chiave possa essere maiuscola o minuscola
   sviluppata come un if_the_else in modo da comportarsi in maniera differente
   in base alla lettera da cifrare passatole
   I numeri utilizzati nelle operazioni corrispondono a dei caratteri in codice ASCII */

vigenere_lettera(T, C, U) :-
       (U1 is T + C, U1 > 90, T < 91),
       U is U1 - 26,
       !.

vigenere_lettera(T, C, U) :-
       (U1 is T + C, U1 >122, T > 96),
       U is U1 - 26,
       !.

vigenere_lettera(T, C, U) :-
       U is T + C.


/*      DECIFRATURA    */

/* funzione che chiede all'utente di inserire la chiave di decifratura e il testo da decifrare
   successivamente chiama la funzione per decifrare il messaggio */

decifrare :-
       write('Hai scelto di decifrare un messaggio'), nl,
       write('NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""'), nl,
       write('Inserisci la chiave di decodifica:'), nl,
       validazione_input(Chiave), nl,
       write('Inserisci il messaggio che vuoi decifrare:'), nl,
       validazione_input(Messaggio),
       if_lunghezza(Messaggio, Chiave, NuovaChiave),
       unvigenere(Messaggio, NuovaChiave, Decifrato),
       write('\nIl messaggio decifrato e\':\t'),
        write(Decifrato).


/* funzione che decripta un messaggio tramite una chiave */

unvigenere(Testo, Chiave, Cifrato) :-
       nonvar(Testo),
       nonvar(Chiave),
       unvigenere_lista(Testo, Chiave, L1),
       name(Cifrato, L1).


/* funzione che controlla se la lettera che sta andando a decriptare è uno spazio:
      -se fosse uno spazio non verrebbe decifrata e si procede con quelle successive
      -se fosse una lettera si procede con la decifratura e poi si procede con le lettere successive
   funziona tramite ricorsione */

unvigenere_lista([], [], []).
unvigenere_lista([TESTA|CODA], [TESTA1|CODA1], [TESTA2|CODA2]) :-
       if_spazio(TESTA,
       unvigenere_lista_lettera([TESTA|CODA], [TESTA1|CODA1], [TESTA2|CODA2]),
       unvigenere_lista_spazio([TESTA|CODA], CODA1, [TESTA2|CODA2])).
```

```
/* funzione che viene utilizzata quando una lettera risulta essere uno spazio
       (sia del testo che della chiave)
   restituisce la lettera originale o uno spazio in base al suo utilizzo
   e procede con la decifratura delle lettere successive */

unvigenere_lista_spazio([], _, []).
unvigenere_lista_spazio([TESTA|CODA], CODA1, [TESTA2|CODA2]) :-
       TESTA2 is TESTA,
       unvigenere_lista(CODA,CODA1, CODA2).


/* funzione che viene utilizzata quando la lettera in decifratura non è uno spazio
   si verifica se la lettera della chiave è uno spazio
   - in caso fosse uno spazio si chiama la funzione unvigenere_lista_spazio
   - in caso non fosse uno spazio si chiama unvigenere_lettera per la decifratura della lettera
     e unvigenere_lista per la decifratura delle lettere successive */

unvigenere_lista_lettera([], [], []).
unvigenere_lista_lettera([TESTA|CODA], [TESTA1|CODA1], [TESTA2|CODA2]) :-
       if_spazio(TESTA1,
       (if_Mm(TESTA1, maiuscola(TESTA1, TESTA3), minuscola(TESTA1, TESTA3)),
       unvigenere_lettera(TESTA, TESTA3, TESTA2),
       unvigenere_lista(CODA,CODA1, CODA2)),
       unvigenere_lista_spazio([TESTA|CODA], CODA1, [TESTA2|CODA2])).


/* funzione che decifra una lettera in base a una chiave
   una lettera maiuscola rimarrà maiuscola mentre una minuscola rimmarrà minuscola nonostante
   la chiave possa essere maiuscola o minuscola
   sviluppata come un if_the_else in modo da comportarsi in maniera differente in base alla lettera
   da decifrare passatole
   I numeri utilizzati nelle operazioni corrispondono a dei caratteri in codice ASCII  */

unvigenere_lettera(T, C, U) :-
       (U1 is T - C, U1 <65, T < 91),
       U is U1 + 26,
       !.

unvigenere_lettera(T, C, U) :-
       (U1 is T - C, U1 < 97, T > 96),
       U is U1 + 26,
       !.

unvigenere_lettera(T, C, U) :-
       U is T - C.
```

5 Testing del programma

Test Haskell 1:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

1
È stato scelto di cifrare un messaggio

Scivere la chiave tra ""
" "

Scivere il testo da cifrare tra ""
" "

Il testo cifrato è:
```

Test Haskell 2:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

2
È stato scelto di decifrare un messaggio

Scivere la chiave tra ""
" "

Scivere il testo da decifrare tra ""
" "

Il testo decifrato è:
```

Test Haskell 3:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

a

Errore
Nb. Inserire solo numeri interi:

1
È stato scelto di cifrare un messaggio

Scivere la chiave tra ""
"chiave"

Scivere il testo da cifrare tra ""
"prova"

Il testo cifrato è:
rywvv
```

Test Haskell 4:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

2
È stato scelto di decifrare un messaggio

Scivere la chiave tra ""
"chiave"

Scivere il testo da decifrare tra ""
"rywvv"

Il testo decifrato è:
prova
```

Test Haskell 5:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

0

Programma terminato
Fate ripartire il programma se desiderate continuare
```

Test Haskell 6:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

1
È stato scelto di cifrare un messaggio

Scivere la chiave tra ""
"Chi4ve"

Errore
Inserire solo lettere MAIUSCOLE, minuscole e spazi:

"Ch%dfw"

Errore
Inserire solo lettere MAIUSCOLE, minuscole e spazi:

"Chiave"

Scivere il testo da cifrare tra ""
"messaggio"

Il testo cifrato è:
olasvkipw
```

Test Haskell 7:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

x

Errore
Nb. Inserire solo numeri interi:

2
È stato scelto di decifrare un messaggio

Scivere la chiave tra ""
"chiave DI codifica"

Scivere il testo da decifrare tra ""
"Pro$a"

Errore
Inserire solo lettere MAIUSCOLE, minuscole e spazi:

"vdf5gt"

Errore
Inserire solo lettere MAIUSCOLE, minuscole e spazi:

"messaggio"

Il testo decifrato è:
kxksfcgfg
```

Test Haskell 8:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

1
È stato scelto di cifrare un messaggio

Scivere la chiave tra ""
"passWord"

Scivere il testo da cifrare tra ""
"Esame di Programmazione Logica e Funzionale"

Il testo cifrato è:
tssea ul pjgcfrpbarakbv aoyayo h fmfvwfqplw
```

Test Haskell 9:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

2
È stato scelto di decifrare un messaggio

Scivere la chiave tra ""
"passWord"

Scivere il testo da decifrare tra ""
"tssea ul pjgcfrpbarakbv aoyayo h fmfvwfqplw"

Il testo decifrato è:
esame di programmazione logica e funzionale
```

Test Haskell 10:

```
Questo programma Haskell contiene al suo interno il cifrario di Vigenère
Si può scegliere di cifrare o decifrare un messaggio tramite una qualsiasi chiave.

Vengono accettate sia lettere maiscole che minuscole,
tuttavia nella fase di cifratura o decifratura verranno tutte restituite in minuscolo.

Scivere:  1       per cifrare una parola o un messaggio
Scivere:  2       per decifrare una parola o un messaggio
Scivere un qualsiasi altro numero per terminare il programma

!

Errore
Nb. Inserire solo numeri interi:

è

Errore
Nb. Inserire solo numeri interi:

1
È stato scelto di cifrare un messaggio

Scivere la chiave tra ""
"Cia/f"

Errore
Inserire solo lettere MAIUSCOLE, minuscole e spazi:

"Ciao"

Scivere il testo da cifrare tra ""
"Testo da cifrare"

Il testo cifrato è:
vmshq do kittirs
```

Test Prolog 1:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
1.

Hai scelto di cifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
esempio: "Chiave o Testo"
Inserisci la chiave di codifica:
" ".

Inserisci il messaggio che vuoi cifrare:
" ".

Il messaggio cifrato e':

(63 ms) yes
```

Test Prolog 2:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
2.

Hai scelto di decifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
Inserisci la chiave di decodifica:
" ".

Inserisci il messaggio che vuoi decifrare:
" ".

Il messaggio decifrato e':

(31 ms) yes
```

Test Prolog 3:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
1.

Hai scelto di cifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
esempio: "Chiave o Testo"
Inserisci la chiave di codifica:
"chiave".

Inserisci il messaggio che vuoi cifrare:
"prova".

Il messaggio cifrato e':        rywvv

(32 ms) yes
```

Test Prolog 4:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
2.

Hai scelto di decifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
Inserisci la chiave di decodifica:
"chiave".

Inserisci il messaggio che vuoi decifrare:
"rywvv".

Il messaggio decifrato e':      prova

(47 ms) yes
```

Test Prolog 5:

```
(47 ms) yes
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
1.

Hai scelto di cifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
esempio: "Chiave o Testo"
Inserisci la chiave di codifica:
"Chi4ve".

input non valido, inserire solamente lettere maiuscole, minuscole e spazi:
"123".

input non valido, inserire solamente lettere maiuscole, minuscole e spazi:
"chiave".

Inserisci il messaggio che vuoi cifrare:
"23cjiab".

input non valido, inserire solamente lettere maiuscole, minuscole e spazi:
"messaggio".

Il messaggio cifrato e':        olasvkipw

(265 ms) yes
```

Test Prolog 6:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
4.

input non valido, inserire solamente 1 o 2:
10.

input non valido, inserire solamente 1 o 2:
2.

Hai scelto di decifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
Inserisci la chiave di decodifica:
"password".

Inserisci il messaggio che vuoi decifrare:
"LaurA E SimonE".

Il messaggio decifrato e':      WaczE N DiuwrQ

(172 ms) yes
```

Test Prolog 7:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
!.

input non valido, inserire solamente 1 o 2:
à.

input non valido, inserire solamente 1 o 2:
1.

Hai scelto di cifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
esempio: "Chiave o Testo"
Inserisci la chiave di codifica:
"Laura & Simone".

input non valido, inserire solamente lettere maiuscole, minuscole e spazi:
"Laura E Simone".

Inserisci il messaggio che vuoi cifrare:
"Come va??".

input non valido, inserire solamente lettere maiuscole, minuscole e spazi:
"Come va".

Il messaggio cifrato e':        Nogv ve

(297 ms) yes
```

Test Prolog 8:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
1.

Hai scelto di cifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
esempio: "Chiave o Testo"
Inserisci la chiave di codifica:
"Esame".

Inserisci il messaggio che vuoi cifrare:
"Programmazione Logica e Funzionale".

Il messaggio cifrato e':        Tjosveemmdmgnq Pgguge e Jyfzusrslq

(234 ms) yes
```

Test Prolog 9:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
a.

input non valido, inserire solamente 1 o 2:
2.

Hai scelto di decifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
Inserisci la chiave di decodifica:
"Esame".

Inserisci il messaggio che vuoi decifrare:
"Tjosveemmdmgnq Pgguge e Jyfzusrslq".

Il messaggio decifrato e':      Programmazione Logica e Funzionale

(157 ms) yes
```

Test Prolog 10:

```
| ?- main.

Questo programma Prolog contiene al suo interno il cifrario di Vigenere
Si puo scegliere se cifrare o decifrare un messaggio secondo una chiave:
Scrivere 1 per cifrare una parola o un messaggio
Scrivere 2 per decifrare una parola o un messaggio
1.

Hai scelto di cifrare un messaggio
NB. ricordarsi di scrivere la chiave e il messaggio racchiusi tra ""
esempio: "Chiave o Testo"
Inserisci la chiave di codifica:
"c".

Inserisci il messaggio che vuoi cifrare:
"Messaggio da cifrare".

Il messaggio cifrato e':        Oguuciikq fc ekhtctg

(453 ms) yes
```