# ***Progetto Assembly RISC-V*** per il Corso di Architetture degli Elaboratori –A.A. 2021/2022 –

# Messaggi in Codice

Autore: Nabiollah Tavakkoli
Numero di matricola: 5953800
Data di consegna: 31/05/2022

E-mail: nabiollah.tavakkoli@stud.unifi.it

## *Il campo **.data***

Contiene 4 elementi principali;

- La stringa "*myplaintext*":
  - che contiene il messaggio da cifrare o decifrare
  - Usato dagli algoritmi ***A, B, D, E*** (ma non ***C***), basta
    - Leggere un carattere della stringa alla volta
    - Modificarlo in base all'algoritmo dato (oppure applicare l'algoritmo)
    - Salvare tale carattere cifrato / decifrato nella sua posizione data in stringa
- La stringa "*mysypher*":
  - Che contiene l'insieme delle chiavi, dove ciascuna chiave indica l'applicazione dell'algoritmo associato a tale chiave
- La stringa "chiave_a_Blocchi":
  - Che contiene la chiave_a_Blocchi, utilizzata soltanto dall'algoritmo C
- ***Un indirizzo Fisso (0xFFF):***
  - Che viene utilizzato **soltanto** dall'algoritmo ***C (Cifratura Occorrenze),*** dove abbiamo bisogno di un'area di memoria di dimensione variabile;
    - cioè a partire da un indirizzo fisso vengono memorizzati i caratteri del messaggio cifrato o decifrato, dove abbiamo bisogno:
      - ♦ di più area di memoria per la cifratura del messaggio,
      - ♦ di meno area di memoria per la decifratura del messaggio già cifrato,

Ma contiene anche le stringhe che vengano direttamente stampate e utilizzate per indicare *la chiave corrente, la stringa corrente, la stringa cifrata, la stringa decifrata, lo spazio, la fine.*

# *metodi generali*

- *main*:
  Si esegue caricando
  - L'indirizzo alla testa della stringa "*myplaintext*" (da cifrare/decifrare) in ***a2***
  - L'indirizzo alla testa della stringa "*mysypher*" (contente le chiavi) in ***s0***
  - caricando un indirizzo di memoria in ***a3***,
    - utilizzato dall'algoritmo *"C_cifraturaOccorrenze*"

- *cifratura:*
  - usato per eseguire tutti gli algoritmi chiamati nella fase di cifratura

- *decifratura:*
  - usato per eseguire tutti gli algoritmi chiamati nella fase di decifratura

- *carica_chiave:*
  - Carica le chiavi della stringa "*mysypher*" una alla volta in un registro, per poter stamparle subito dopo
  - Memorizza l'indirizzo alla testa della stringa "*mysypher*" in ***a4***
    - ***a4*** poi viene passato ai vari metodi come argomento contenete l'indirizzo alla testa della stringa da cifrare o decifrare
  - Salta al metodo *print_correnti*, per stampare la chiave e la stringa corrente
  - Se ***a5*** non contiene il valore ZERO, allora salta al metodo *print_dopo_cifraturaDecifratura* per poter stampare la stringa modificata (dopo ogni cifratura o decifratura)
  - se dopo la cifratura ***a1*** contiene il valore ZERO, allora si inizia la fase di decifratura
  - se dopo la decifratura ***a1*** contiene il valore ZERO, allora il programma termina, stampando "**FINE**"
  - In ogni altro caso, chiama il metodo (l'algoritmo) corrispondente alla chiave (al carattere letto dalla stringa "mysypher")

- length:
  - permette di ottenere la lunghezza di una stringa (compreso anche la fine_della_stringa)
  - tale valore viene memorizzato (restituito) in ***a0***

- print_correnti:
  - permette di stampare
    - la chiave corrente (già memorizzato nel registro ***a1***) e
    - la stringa corrente

- print_dopo_cifraturaDecifratura:
  - viene utilizzato per stampare la stringa dopo la cifratura oppure decifratura

## *Registri importanti*

- s0: contiene l'indirizzo alla testa della stringa "mysypher"
- s1: contiene la chiave per eseguire lo shift nell'algoritmo di "*cifrario a sostituzione*"
- s2: contiene la lunghezza della stringa "chiave_a_Blocchi"
- s3: contiene l'indirizzo alla testa della stringa "chiave_a_Blocchi"
- a2: contiene l'indirizzo alla testa della stringa "myplaintext"
- a3: contiene il valore 0xFFF (***li a3, 0xFFF***)
  - usato soltanto dall'algoritmo *C_cifraturaOccorrenze*
  - il suo valore cambia dopo l'applicazione dell'algoritmo *C_cifraturaOccorrenze*
- a4: contiene l'indirizzo alla testa della stringa da cifrare oppure decifrare
  - è il parametro di ogni metodo
  - viene passato ad ogni algoritmo (***add a4, a2, zero*** oppure ***add a4, a3, zero***)
- a5: contiene l'indice di ciascun carattere della stringa "mysypher"
  - usato per caricare ciascun carattere della stringa (in Cifratura e Decifratura)
  - non deve essere modificato negli altri metodi
- a6: indica lo stato in cui si trova il metodo (di cifratura o decifratura):
  - se a6 = 0, allora andiamo a cifrare
  - se a6 = 1, allora andiamo a decifrare

## *1. A_cifrarioSostituzione*

L'algoritmo sarà chiamato sia per la cifratura che la decifratura della stringa "myplaintext". Esso viene eseguito modificando soltanto il codice ASCII delle lettere minuscole e maiuscole della stringa in base alle formule date, dove: ***cod(x)*** è la codifica ASCII standard decimale di un carattere ***x***

**lettera maiuscola (A-Z, ASCII 65-90 compresi): *65 + [(cod(X) - 65) + K] mod 26***

**lettera maiuscola (a-z, ASCII 97-122 compresi*): 97 + [(cod(X) - 97) + K] mod 26***

Viene suddiviso in 4 metodi principali:

1.1 "*cifrarioSostituzione_loop*"

- Poiché l'algoritmo si applica soltanto alle lettere maiuscole e minuscole e tutte le altre restano invariati, allora abbiamo bisogno di leggere la stringa "myplaintext" carattere per carattere ed eseguire un confronto
  - Se tale carattere rappresenta una lettera minuscola oppure maiuscola, allora la cifratura/decifratura continua
  - Altrimenti, si ritorna al metodo, caricando il prossimo carattere da confrontare

- ***Funzionamento:***
  - Per ottenere il carattere da cifrare o decifrare salta al metodo "*MR_RM*"
    - Il carattere viene salvato nel registro ***a0***, ed il suo indice nel registro ***a1***
  - Si esegue un confronto, utilizzando il codice ASCII standard decimale del carattere
    - Se tale carattere rappresenta una lettera maiuscola/minuscola, allora la cifratura oppure decifratura continua saltando al metodo "*cifrarioSostituzione_CifDecif* "
    - Altrimenti, salta di nuovo al metodo *cifrarioSostituzione_loop*", lasciando la lettera invariata

1.2 "*cifrarioSostituzione_CifDecif*"

- Si esegue il primo passo della formula utilizzata per la cifratura e decifratura, sottraendo dal codice ASCII del carattere il valore 65 (per le lettere minuscole) oppure 97 (per le lettere maiuscole), tale valore è già memorizzato nel registro ***t4***
- E poi, in base al valore contenuto nel registro ***a6*** (che indica lo stato di cifratura o decifratura) abbiamo:
  - Se a6 = 0, il carattere sarà cifrato, sommando la chiave (presente in ***s1***)
  - Se a6 = 1, il carattere sarà decifrato, sottraendo la chiave (presente in ***s1***)
- In ogni caso, salta al metodo "cifrarioSostituzione_CifDecif_continua"

1.3 "cifrarioSostituzione_CifDecif_continua"

- Per poter calcolare ***[(cod(X)-65) + k] % 26*** abbiamo bisogno che il valore ***[(cod(X)-65) + k]*** sia un numero positivo e per fare ciò
  - Salta al metodo "resto_positivo" per ottenere un dividendo positivo
- Si esegue, calcolando il resto della divisione del valore ***[(cod(X)-65) + k]*** rispetto al ***26*** e poi somma tale risultato al valore ***65*** oppure ***97***
- Si salta di nuovo al metodo "*cifrarioSostituzione_loop*"
  - Il carattere modificato (presente nel registro ***a0***) viene salvato in memoria, saltando al metodo "*MR_RM*" e usando l'indice del carattere già memorizzato ***a1***

1.4 “resto_positivo”

- Poiché nel caso della decifratura, il dividendo può risultare negativo, allora questo metodo viene eseguito per sommare il dividendo (negativo) al valore 26 finché il suo resto della divisione rispetto al 26 non risulti positivo.

## *2. B_cifrarioBlocchi*

L’algoritmo viene utilizzato sia nel caso di cifratura che la decifratura, e viene suddiviso in 4 metodi principali:

2.1 Inizialmente

- Salta al metodo “length” per poter ottenere la lunghezza della stringa “chiave_a_Blocchi” che poi viene utilizzato per poter estrarre la chiave corretta da sommare a ciascun carattere della stringa “myplaintext”, Cioè;
  - L’indice della chiave da estrarre nella stringa “chiave_a_Blocchi” è dato dalla formula ***i % key_length***, dove:
    - ***i*** è l’indice del carattere nella stringa “myplaintext”
    - ***key_length*** è la lunghezza della stringa “chiave_a_Blocchi”

2.2 cifrarioBlocchi_key_char:

- In questo metodo vengono estratti, il carattere da cifrare o decifrare della stringa “myplaintext” e la chiave della stringa “chiave_a_Blocchi” e vengono memorizzati rispettivamente nei registri ***a0 e t4***

2.3 distinzione_cifratura_decifratura:

- tale metodo viene utilizzato per indicare lo stato di cifratura o decifratura in base al contenuto del registro ***a6***
  - Se a6 = 1, allora salta al metodo “decifrario_a_Blocchi”
  - Se a6 = 0, si esegue in base alla formula data, cifrando il carattere e poi salta a “incremento_indice_Key_char” per poter incrementare il contatore della stringa e caricare il prossimo carattere da cifrare

$$cb_i = 32 + (cod(b_{ij}) + cod(key_j)) \bmod 96$$

2.4 “decifrario_a_Blocchi”:

- Viene eseguito soltanto nel caso della decifratura
- Si esegue in base alla formula data

$$cb_i - 32 - cod(key_j) + 96*k$$

  - Invece di moltiplicare il valore ***96*** per una costante ***k***, salta al metodo “divisore_corretto”
    - Sommando il valore ottenuto dall’equazione [$cb_i + 32 - cod(key_j)$] al valore ***96***, finché il codice ASCII del carattere non risulti MAGGIORE O UGUALE a ***32***, che indica il minimo valore del codice ASCII di un carattere nell’intervallo dato:

$$(32 =< cod(x) <= 127)$$

### *3. C_cifraturaOccorrenze*

Sono state implementate due metodi separati per la cifratura e la decifratura:

3.1 Cifratura:

3.1.1. Inizialmente

- Salta al metodo "carica_del_carattere" per carica il primo carattere della stringa "myplaintext" nel registro ***a0*** ed il suo indice nel registro ***a1***
- Poiché un carattere può avere più di una occorrenza, allora per evitare le ridondanze e caricamento dei duplicati, il primo carattere viene memorizzato anche nel registro ***s4*** (che inizialmente assume valore "-1", per evitare il primo confronto), il contenuto del registro ***s4***:
  - Sovrascrive ogni carattere che risulta uguale al contenuto del registro ***a0***, e
  - Impedisce il caricamento dei caratteri già visitati e risultati uguale ai precedenti ***a0***

3.1.2. "carica_del_carattere"

- Viene utilizzato per caricare un carattere (della stringa "myplaintext") ed il suo indice rispettivamente nei registri ***a0 e a1***
  - Se ***a0*** contiene il valore ZERO, l'algoritmo termina ponendo l'ultimo carattere della stringa risultante uguale a ZERO, che indica la fine della stringa
- Confronta il codice ASCII del carattere memorizzato nel registro ***a0*** con il codice ASCII del carattere presente nel registro ***s4*** (per evitare le ridondanze)

3.1.3. "continua_carica_dei_caratteri"

- I caratteri della stringa risultante vengano memorizzati a partire da un nuovo indirizzo, già memorizzato nel registro ***a3***

3.1.4. "carica_delle_occorrenza_loop"

- Carica ciascun carattere della stringa "myplaintext" a partire dall'indice memorizzato nel registro ***a1***
  - Se il carattere ottenuto è zero, vuol dire non ci sono oppure non ci sono di più le occorrenze del carattere nella stringa, allora salta al metodo "end_questa_occorrenza" per controllare le occorrenze di altri caratteri
  - Se il carattere è diverso dal carattere memorizzato nel registro ***a0***, allora salta al metodo "incremento_indice_loop" per poter caricare il prossimo carattere da confrontare con ***a0***
  - Altrimenti,
    - Il contatore della stringa risultante incrementa di un'unità e in tale posizione viene memorizzato il codice ASCII del carattere trattino (***ASCII (-) = 45***)
    - Poiché per le occorrenze si devono usare tanti bytes quante sono le cifre del numero (indice), allora si esegue un confronto per poter conoscere le cifre maggiore di 9 (con più di una cifra):
      - Se la cifra è minore o uguale al numero puro 9, allora va a essere modificato e memorizzato in memoria
      - Altrimenti, salta al metodo "indiciGrandi" per poter trasformare tale cifra in un insieme di cifre minore o uguale al 9

3.1.5. “indiciGrandi”

- Memorizza l’indice nel registro ***t5***,
  - consideriamo una cifra maggiore di 9 come una sequenza delle cifre minore o uguale a 9
- Finché ***t5*** non è uguale a zero, si esegue
  - Calcolando il resto della divisione del numero (indice) contenuto nel ***t5*** rispetto al ***10*** (ottenendo la prima cifra della sequenza delle cifre) e memorizzare tale valore nel registro ***t1***
  - Saltando al metodo “carica_delle_cifre_nelloStack” per memorizzare tale cifra nello stack
  - Poi ritornando dal metodo “carica_delle_cifre_nelloStack” si divide ***t5*** per ***10***, per eliminare la cifra memorizzata nello stack dalla sequenza delle cifre
- Se ***t5*** risulta uguale a zero, allora va caricare, una alla volta, le cifre già memorizzate nello stack, in memoria

3.1.6. “carica_delle_cifre_nelloStack”

- Alloca uno spazio nello stack, diminuendo di 4 unità la dimensione del registro ***sp***
- Memorizza la cifra contenuta nel registro ***t1*** (che indica un numero puro [0, 9]) alla testa dello stack.
- Incrementa il contenuto della stringa ***t6*** di un’unità ad ogni inserimento nello stack
  - ***t6*** indica la lunghezza dello stack (il numero degli elementi memorizzati nello stack).
- Ritorna al chiamante

3.1.7. “carica_delle_cifre_inMemoria”

- Finché il contenuto del registro ***t6*** non risulti uguale a zero
  - Diminuisce la lunghezza dello stack di un’unità
  - Carica l’ultima cifra memorizza nello stack nel registro ***t1***
  - Poi somma:
    - la cifra contenuta nel registro ***t1*** (che indica un numero puro compreso nell’intervallo ***[0, 9]***)
    - al contenuto del registro ***s8*** (***s8 = ASCII (0) = 48***), per ottenere il codice ASCII della cifra, che viene memorizzata nel registro ***s6***
  - Alla fine, salta al metodo “carica_inMemoria” per memorizzare il codice ASCII di ciascuna cifra nella posizione corretta della stringa risultante
    - Data la proprietà ***LIFO*** dello stack le cifre vengono estratti e memorizzati nell’ordine corretto (da più significativa a meno significativa)

3.2 Decifratura:

3.2.1. “decifOc_caricaCarattere”

- Inizialmente salta al metodo “carica_del_carattere_decif” per poter caricare il primo carattere della stringa (che non rappresenta un indice) nel registro ***a0*** ed il suo indice (la sua posizione nella stringa data) nel registro ***a1***
- Dopo incrementa il valore ***a1*** di un’unità per evitare il primo confronto (perché il carattere successivo sarà sempre un trattino (come: ***A-23…***)

3.2.2. “decifOc_caricaOccorrenze”

- Inizia a caricare il carattere successivo in un registro (***t3***), se il prossimo carattere è:
  - ***Lo zero:***
    - indica la fine dell’algoritmo
    - salta al metodo “termine_occorrenza”, sommando al contenuto del registro ***a3*** (che indica l’indirizzo alla testa della stringa risultante) il contenuto del registro ***t5*** (che indica la lunghezza della stringa risultante)
      - Alla fine, somma anche al risultato ***a3 + t5*** il valore ***2*** per essere sicuro che ci siano abbastanza spazi tra l’indirizzo alla testa della stringa risultante ed il nuovo indirizzo ([***a3 + t5] + 2***)
      - Poiché le occorrenze sono elencate subito dopo il carattere quindi, lo zero certamente indica “la fine della stringa”
  - ***Uno spazio:***
    - indica che tale carattere non ha più delle occorrenze
    - Salta al metodo “carica_carattere_successivo”, incrementando il contatore della stringa (***a1***) di un’unità
  - ***Un trattino:***
    - Indica che il carattere (già caricato in ***a0***) ha almeno un’altra occorrenza, allora va a trova l’indice corrispondente al carattere
    - Salta al metodo “incremento_contatore”, incrementando il contatore della stringa (***a1***) di un’unità

3.2.3. “cercaIndice”

- Dato il metodo della cifratura, un’occorrenza può avere una oppure più di una cifra e le sue cifre sono memorizzate in codice ASCII e possono apparire soltanto tra:
  - Due trattini, oppure
  - Un trattino ed uno spazio, oppure
  - Un trattino e la fine della stringa
- Allora finché non si trova il valore ***ZERO*** (che indica la fine stringa) oppure il valore ***32*** (che indica uno spazio) oppure il valore ***45*** (che indica un trattino):
  - Per ottenere le cifre dell’indice come valore puro (cioè come un numero compreso nell’intervallo ***[0, 9]***), il valore ***48*** (presente nel registro ***s8***), viene sottratto dal codice ASCII dell’indice (presente nel registro ***t3***)
  - Tale valore puro viene memorizzato alla testa dello stack
  - Incrementa il valore del registro ***t6***, di un’unità (che indica la lunghezza dello stack)
  - Se trova uno dei valori ***0/32/45*** allora va a calcolare l’indice corrispondente, saltando al metodo “calcolaIndice”

3.2.4. “calcolaIndice”:

- Inizialmente diminuisce il contatore della stringa (***a1***) di un’unità, poiché l’ultimo carattere caricato conteneva uno dei valori ***0/32/45***
- Data la proprietà ***LIFO*** dello stack, l’ultima cifra memorizzata sarà la cifra meno significativa, quindi a partire dalla testa dello stack
  - Le cifre vengono estratte e memorizzate nel registro ***t4***
  - La cifra viene moltiplicata per un multiplo di ***10 (1, 10, 100, …)*** e sommata alla cifra precedente, formando un indice come un valore puro
  - Ad ogni passo diminuisce la lunghezza dello stack (presente nel registro ***t6***) di un’unità
  - Il metodo termina, quando ***t6*** indica il valore zero, allora salta al metodo “calcolaIndice_loop_end”, incrementando il contenuto del registro ***t5*** di un’unità che indica la lunghezza della stringa risultante e poi tale carattere già memorizzato nel registro ***a0***, viene memorizzato nella posizione indica dall’indice, cioè: ***[(t3 - 1) + nuovo_indirizzo]***
    - ***t3*** diminuisce di un’unità poiché le occorrenze iniziano da ***1*** ma non ***0***

### *4. D_dizionario*

Questo algoritmo viene utilizzato sia per la cifratura che la decifratura, dove:

- La conversione da minuscolo a MAIUSCOLO delle lettere dell'alfabeto, si ottiene: ***Sottraendo il valore 187*** dal codice ASCII di ciascuna lettera, tale valore si ottiene nel modo seguente, dobbiamo:
  - Convertire la lettera minuscola in MAIUSCOLA, usando la formula:

    ***MAIUSCOLO(X) = minuscolo(x) - 32***
  - Sottrarre il codice ASCII della lettera MAIUSCOLA dal codice ASCII della prima lettera MAIUSCOLA (ASCII(A) = 65)

    ***MAIUSCOLO(X) - 65***
  - Eseguire la sottrazione: ***25 – [MAIUSCOLO(X) - 65]***
  - Eseguire la somma: ***65 + [25 – [MAIUSCOLO(X) - 65]]***
  - Se facciamo tutti i calcoli uno dopo l'altro, si ottiene:

    ***65 + [25 – [(minuscolo(x) - 32) - 65]] = 65 + 25 - minuscolo(x) + 32 + 65***

    ***= 187 - minuscolo(x)***

- La conversione da MAIUSCOLO a minuscolo, si ottiene, eseguendo gli stessi passi utilizzati per la conversione da minuscolo a MAIUSCOLO, ma
  - Invece il valore 65, utilizziamo il codice ASCII della prima lettera minuscola (ASCII(a) = 97), e
  - Per ottenere il codice ASCII della lettera minuscola, invece di sottrarre il valore 32, dovrebbe sommare il valore 32 al codice ASCII della lettera MAIUSCOLA
  - Ottenendo alla fine, la stessa equazione:

    ***97 + [25 – [(minuscolo(x) + 32) - 97]] = 97 + 25 - minuscolo(x) - 32 + 97***

    ***= 187 - MAIUSCOLO(X)***

- La codifica del numero, si ottiene, sottraendo il valore ***105*** dal codice ASCII di ciascun numero, tale valore si ottiene nel modo seguente, dobbiamo:
  - Eseguire l'equazione ***ASCII (9) – ASCII (numero),*** per ottenere un valore puro compreso nell'intervallo [0, 9], e poi
  - Sommare il risultato ottenuto (precedentemente) al valore ***48 (ASCII (0) = 48***), per ottenere il codice ASCII di un numero tra 0 e 9
  - Se facciamo tutti i calcoli uno dopo l'altro, sia per la cifratura che la decifratura, si ottiene la stessa equazione:

    - Cifratura:

      ***48 + [ASCII (9) – ASCII (numero)] = 48 + [57 – ASCII (numero)] =***

      ***= 105 - ASCII (numero)]***
    - Decifratura:

      ***ASCII (9) – [ASCII (numero) – 48] = 105 - ASCII (numero)]***

### 5. *E_inversione*

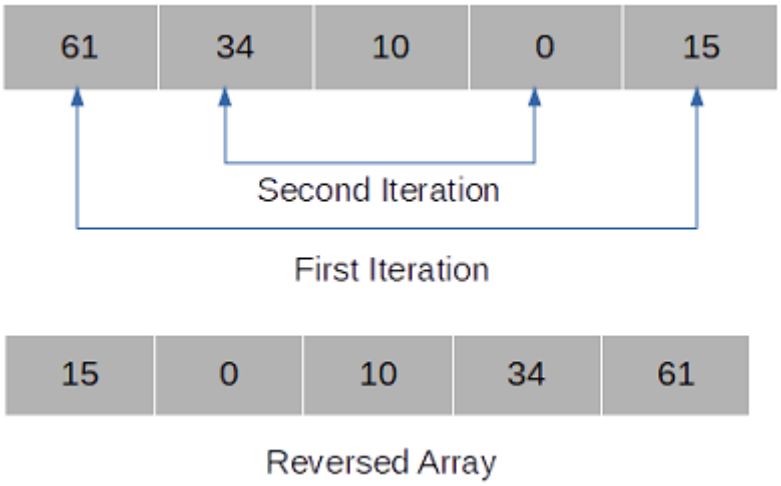
Questo algoritmo viene utilizzato sia per la cifratura che la decifratura, Praticamente viene utilizzato per invertire sia la stringa “myplaintext” che la stringa “mysypher” (per poter ottenere le chiavi nell’ordine inverso nella fase di decifratura)**.**

L’algoritmo si esegue:

- Chiamando il metodo “length”, per calcolare la lunghezza della stringa
  - Diminuisce la lunghezza ottenuta di un’unità (corrispondente alla fine della stringa)
- Caricando 2 byte della stringa (due caratteri), posti nelle posizioni opposte (per esempio, il primo e l’ultimo) in due registri temporanei ***t3, t4***
- Eseguendo uno swap

Algoritmo termina

- quando i due indici ***i, j*** (che indicano rispettivamente l’inizio e la fine della stringa) si incrociano



***Esempi***

```
10 myplaintext: .string "sempio di messaggio criptato -1ASD"
11 mysypher: .string "ABCDE"
12 chiave_a_Blocchi: .string "OLE"
```

Console

```
chiave corrente = A
stringa corrente = sempio di messaggio criptato -1ASD
stringa cifrata = mygjci xc gymmuaaci wlcjnuni -1UMX

chiave corrente = B
stringa corrente = mygjci xc gymmuaaci wlcjnuni -1UMX
stringa cifrata = |%lyon/$h/s~|yzpmhx,|{oo}!sx,2@aRg

chiave corrente = C
stringa corrente = |%lyon/$h/s~|yzpmhx,|{oo}!sx,2@aRg
stringa cifrata = |-1-13-21 %-2 l-3 y-4-14 o-5-23-24 n-6 /-7-10 $-8 h-9-18 s-11-27 ~-12 z-15 p-16 m-17 x-19-28 ,-20-29 {-22 }-25 !-26 2-30 @-31 a-32 R-33 g-34

chiave corrente = D
stringa corrente = |-1-13-21 %-2 l-3 y-4-14 o-5-23-24 n-6 /-7-10 $-8 h-9-18 s-11-27 ~-12 z-15 p-16 m-17 x-19-28 ,-20-29 {-22 }-25 !-26 2-30 @-31 a-32 R-33 g-34
stringa cifrata = |-8-86-78 %-7 O-6 B-5-85 L-4-76-75 M-3 /-2-89 $-1 S-0-81 H-88-72 ~-87 A-84 K-83 N-82 C-80-71 ,-79-70 {-77 }-74 !-73 7-69 @-68 Z-67 i-66 T-65

chiave corrente = E
stringa corrente = |-8-86-78 %-7 O-6 B-5-85 L-4-76-75 M-3 /-2-89 $-1 S-0-81 H-88-72 ~-87 A-84 K-83 N-82 C-80-71 ,-79-70 {-77 }-74 !-73 7-69 @-68 Z-67 i-66 T-65
stringa cifrata = 56-T 66-i 76-Z 86-@ 96-7 37-! 47-} 77-{ 07-97-, 17-08-C 28-N 38-K 48-A 78-~ 27-88-H 18-0-S 1-$ 98-2-/ 3-M 57-67-4-L 58-5-B 6-O 7-% 87-68-8-|

chiave corrente = E
stringa corrente = 56-T 66-i 76-Z 86-@ 96-7 37-! 47-} 77-{ 07-97-, 17-08-C 28-N 38-K 48-A 78-~ 27-88-H 18-0-S 1-$ 98-2-/ 3-M 57-67-4-L 58-5-B 6-O 7-% 87-68-8-|
stringa decifrata = |-8-86-78 %-7 O-6 B-5-85 L-4-76-75 M-3 /-2-89 $-1 S-0-81 H-88-72 ~-87 A-84 K-83 N-82 C-80-71 ,-79-70 {-77 }-74 !-73 7-69 @-68 Z-67 i-66 T-65

chiave corrente = D
stringa corrente = |-8-86-78 %-7 O-6 B-5-85 L-4-76-75 M-3 /-2-89 $-1 S-0-81 H-88-72 ~-87 A-84 K-83 N-82 C-80-71 ,-79-70 {-77 }-74 !-73 7-69 @-68 Z-67 i-66 T-65
stringa decifrata = |-1-13-21 %-2 l-3 y-4-14 o-5-23-24 n-6 /-7-10 $-8 h-9-18 s-11-27 ~-12 z-15 p-16 m-17 x-19-28 ,-20-29 {-22 }-25 !-26 2-30 @-31 a-32 R-33 g-34

chiave corrente = C
stringa corrente = |-1-13-21 %-2 l-3 y-4-14 o-5-23-24 n-6 /-7-10 $-8 h-9-18 s-11-27 ~-12 z-15 p-16 m-17 x-19-28 ,-20-29 {-22 }-25 !-26 2-30 @-31 a-32 R-33 g-34
stringa decifrata = |%lyon/$h/s~|yzpmhx,|{oo}!sx,2@aRg

chiave corrente = B
stringa corrente = |%lyon/$h/s~|yzpmhx,|{oo}!sx,2@aRg
stringa decifrata = mygjci xc gymmuaaci wlcjnuni -1UMX

chiave corrente = A
stringa corrente = mygjci xc gymmuaaci wlcjnuni -1UMX
stringa decifrata = sempio di messaggio criptato -1ASD

FINE
```

Execution info

| | |
|---|---|
| Cycles: | 19286 |
| Instrs. retired: | 19286 |
| CPI: | 1 |
| IPC: | 1 |
| Clock rate: | 11.43 KHz |

```
10 myplaintext: .string "AMO AssEMBLY 98765"
11 mysypher: .string "ABCDE"
12 chiave_a_Blocchi: .string "OLE"
```

Console

```
chiave corrente = A
stringa corrente = AMO AssEMBLY 98765
stringa cifrata = UGI UmmYGVFS 98765

chiave corrente = B
stringa corrente = UGI UmmYGVFS 98765
stringa cifrata = dSN/ar|eLeRX/E=FB:

chiave corrente = C
stringa corrente = dSN/ar|eLeRX/E=FB:
stringa cifrata = d-1 S-2 N-3 /-4-13 a-5 r-6 |-7 e-8-10 L-9 R-11 X-12 E-14 =-15 F-16 B-17 :-18

chiave corrente = D
stringa corrente = d-1 S-2 N-3 /-4-13 a-5 r-6 |-7 e-8-10 L-9 R-11 X-12 E-14 =-15 F-16 B-17 :-18
stringa cifrata = W-8 h-7 m-6 /-5-86 Z-4 I-3 |-2 V-1-89 o-0 i-88 c-87 v-85 =-84 u-83 y-82 :-81

chiave corrente = E
stringa corrente = W-8 h-7 m-6 /-5-86 Z-4 I-3 |-2 V-1-89 o-0 i-88 c-87 v-85 =-84 u-83 y-82 :-81
stringa cifrata = 18-: 28-y 38-u 48-= 58-v 78-c 88-i 0-o 98-1-V 2-| 3-I 4-Z 68-5-/ 6-m 7-h 8-W

chiave corrente = E
stringa corrente = 18-: 28-y 38-u 48-= 58-v 78-c 88-i 0-o 98-1-V 2-| 3-I 4-Z 68-5-/ 6-m 7-h 8-W
stringa decifrata = W-8 h-7 m-6 /-5-86 Z-4 I-3 |-2 V-1-89 o-0 i-88 c-87 v-85 =-84 u-83 y-82 :-81

chiave corrente = D
stringa corrente = W-8 h-7 m-6 /-5-86 Z-4 I-3 |-2 V-1-89 o-0 i-88 c-87 v-85 =-84 u-83 y-82 :-81
stringa decifrata = d-1 S-2 N-3 /-4-13 a-5 r-6 |-7 e-8-10 L-9 R-11 X-12 E-14 =-15 F-16 B-17 :-18

chiave corrente = C
stringa corrente = d-1 S-2 N-3 /-4-13 a-5 r-6 |-7 e-8-10 L-9 R-11 X-12 E-14 =-15 F-16 B-17 :-18
stringa decifrata = dSN/ar|eLeRX/E=FB:

chiave corrente = B
stringa corrente = dSN/ar|eLeRX/E=FB:
stringa decifrata = UGI UmmYGVFS 98765

chiave corrente = A
stringa corrente = UGI UmmYGVFS 98765
stringa decifrata = AMO AssEMBLY 98765

FINE
```

Execution info

| | |
|---|---|
| Cycles: | 9818 |
| Instrs. retired: | 9818 |
| CPI: | 1 |
| IPC: | 1 |
| Clock rate: | 19.06 KHz |

```
10 myplaintext: .string "myStr0ng p4ssW_%$"
11 mysypher: .string "ACBECDDBE"
12 chiave_a_Blocchi: .string "OLE"
```

Console

```
chiave corrente = A
stringa corrente = myStr0ng p4ssW_%$
stringa cifrata = gsMnl0ha j4mmQ_%$

chiave corrente = C
stringa corrente = gsMnl0ha j4mmQ_%$
stringa cifrata = g-1 s-2 M-3 n-4 l-5 0-6 h-7 a-8  -9 j-10 4-11 m-12-13 Q-14 _-15 %-16 $-17

chiave corrente = B
stringa corrente = g-1 s-2 M-3 n-4 l-5 0-6 h-7 a-8  -9 j-10 4-11 m-12-13 Q-14 _-15 %-16 $-17
stringa cifrata = v96/  2A,R<?%}99/x2D,5<B%w9</m2G,%<E%y96?,9<=6/y2@>2@?%`96C,d<=:/12@B%396F

chiave corrente = E
stringa corrente = v96/  2A,R<?%}99/x2D,5<B%w9</m2G,%<E%y96?,9<=6/y2@>2@?%`96C,d<=:/12@B%396F
stringa cifrata = F693%B@21/:=<d,C69`%?@2>@2y/6=<9,?69y%E<%,G2m/<9w%B<5,D2x/99}%?<R,A2  /69v

chiave corrente = C
stringa corrente = F693%B@21/:=<d,C69`%?@2>@2y/6=<9,?69y%E<%,G2m/<9w%B<5,D2x/99}%?<R,A2  /69v
stringa cifrata = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73

chiave corrente = D
stringa corrente = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73
stringa cifrata = u-8 3-7-82-70-64-28 0-6-81-67-63-51-40-39-27 6-5 %-4-79-61-58-49-37 y-3-48 @-2-77-74 7-1-76-73-55-43-31 8-0 /-89-71-53-41-29 :-88 =-87-69 <-86-68-59-52-47-35
W-85 ,-84-66-57-45-33 x-83 `-80 ?-78-65-36 >-75 B-72-62 v-60 t-56 N-54 D-50 4-46 w-44 C-42 }-38 i-34 z-32  -30 E-26

chiave corrente = D
stringa corrente = u-8 3-7-82-70-64-28 0-6-81-67-63-51-40-39-27 6-5 %-4-79-61-58-49-37 y-3-48 @-2-77-74 7-1-76-73-55-43-31 8-0 /-89-71-53-41-29 :-88 =-87-69 <-86-68-59-52-47-35
W-85 ,-84-66-57-45-33 x-83 `-80 ?-78-65-36 >-75 B-72-62 v-60 t-56 N-54 D-50 4-46 w-44 C-42 }-38 i-34 z-32  -30 E-26
stringa cifrata = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73

chiave corrente = B
stringa corrente = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73
stringa cifrata = U96/B2A96F97H98D9<@,><?2@D2B>2BB2CD2DE2E<2F>%B99/12D97?98G99@9:?9;A,G<B2D=%O9<<>7<>:/>2G97B97E99C9:E9;G,6<E%>96?97G99E9:G9<?,?<=6/
I2@>2B<%K96B98@99?99F9:A9;C,i<=9/82@A2B?2C>2D@2EB%R96E,e<=>/K2A=2B@2E?%M97C,~<><<?</Q2BE%V99B,r<@:/#2CE%D9:B,I<A:/$2DC%,9;@,W<B:/M2EC%.9;H,{<C8

chiave corrente = E
stringa corrente = U96/B2A96F97H98D9<@,><?2@D2B>2BB2CD2DE2E<2F>%B99/12D97?98G99@9:?9;A,G<B2D=%O9<<>7<>:/>2G97B97E99C9:E9;G,6<E%>96?97G99E9:G9<?,?<=6/
I2@>2B<%K96B98@99?99F9:A9;C,i<=9/82@A2B?2C>2D@2EB%R96E,e<=>/K2A=2B@2E?%M97C,~<><<?</Q2BE%V99B,r<@:/#2CE%D9:B,I<A:/$2DC%,9;@,W<B:/M2EC%.9;H,{<C8
stringa cifrata = 8C<{,H;9.%CE2M/:B<W,@;9,%CD2$/:A<I,B:9D%EC2#/:@<r,B99V%EB2Q/<?<<><~,C79M%?E2@B2=A2K/>=<e,E69R%BE2@D2>C2?B2A@28/9=<i,C;9A:9F99?
99@89B69K%<B2>@2I/6=<?,?<9G:9E99G79?69>%E<6,G;9E:9C99E79B79G2>/:><7><<9O%=D2B<G,A;9?:9@99G89?79D21/99B%>F2<E2ED2DC2BB2>B2D@2?
<>,@<9D89H79F69A2B/69U

chiave corrente = E
stringa corrente = 8C<{,H;9.%CE2M/:B<W,@;9,%CD2$/:A<I,B:9D%EC2#/:@<r,B99V%EB2Q/<?<<><~,C79M%?E2@B2=A2K/>=<e,E69R%BE2@D2>C2?B2A@28/9=<i,C;9A:9F99?
99@89B69K%<B2>@2I/6=<?,?<9G:9E99G79?69>%E<6,G;9E:9C99E79B79G2>/:><7><<9O%=D2B<G,A;9?:9@99G89?79D21/99B%>F2<E2ED2DC2BB2>B2D@2?
<>,@<9D89H79F69A2B/69U
stringa decifrata = U96/B2A96F97H98D9<@,><?2@D2B>2BB2CD2DE2E<2F>%B99/12D97?98G99@9:?9;A,G<B2D=%O9<<>7<>:/>2G97B97E99C9:E9;G,6<E%>96?97G99E9:G9<?,?<=6/
I2@>2B<%K96B98@99?99F9:A9;C,i<=9/82@A2B?2C>2D@2EB%R96E,e<=>/K2A=2B@2E?%M97C,~<><<?</Q2BE%V99B,r<@:/#2CE%D9:B,I<A:/$2DC%,9;@,W<B:/M2EC%.9;H,{<C8
```

Execution info

| | |
|---|---|
| Cycles: | 64442 |
| Instrs. retired: | 64442 |
| CPI: | 1 |
| IPC: | 1 |
| Clock rate: | 17.00 KHz |

```
chiave corrente = B
stringa corrente = U96/B2A96F97H98D9<@,><?2@D2B>2BB2CD2DE2E<2F>%B99/12D97?98G99@9:?9;A,G<B2D=%O9<<>7<>:/>2G97B97E99C9:E9;G,6<E%>96?97G99E9:G9<?,?<=6/
I2@>2B<%K96B98@99?99F9:A9;C,i<=9/82@A2B?2C>2D@2EB%R96E,e<=>/K2A=2B@2E?%M97C,~<><<?</Q2BE%V99B,r<@:/#2CE%D9:B,I<A:/$2DC%,9;@,W<B:/M2EC%.9;H,{<C8
stringa decifrata = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73

chiave corrente = D
stringa corrente = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73
stringa decifrata = u-8 3-7-82-70-64-28 0-6-81-67-63-51-40-39-27 6-5 %-4-79-61-58-49-37 y-3-48 @-2-77-74 7-1-76-73-55-43-31 8-0 /-89-71-53-41-29 :-88 =-87-69 <-86-68-59-52-47-35
W-85 ,-84-66-57-45-33 x-83 `-80 ?-78-65-36 >-75 B-72-62 v-60 t-56 N-54 D-50 4-46 w-44 C-42 }-38 i-34 z-32  -30 E-26

chiave corrente = D
stringa corrente = u-8 3-7-82-70-64-28 0-6-81-67-63-51-40-39-27 6-5 %-4-79-61-58-49-37 y-3-48 @-2-77-74 7-1-76-73-55-43-31 8-0 /-89-71-53-41-29 :-88 =-87-69 <-86-68-59-52-47-35
W-85 ,-84-66-57-45-33 x-83 `-80 ?-78-65-36 >-75 B-72-62 v-60 t-56 N-54 D-50 4-46 w-44 C-42 }-38 i-34 z-32  -30 E-26
stringa decifrata = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73

chiave corrente = C
stringa corrente = F-1 6-2-17-29-35-71 9-3-18-32-36-48-59-60-72 3-4 %-5-20-38-41-50-62 B-6-51 @-7-22-25 2-8-23-26-44-56-68 1-9 /-10-28-46-58-70 :-11 =-12-30 <-13-31-40-47-52-64
d-14 ,-15-33-42-54-66 C-16 `-19 ?-21-34-63 >-24 y-27-37 E-39 G-43 m-45 w-49 5-53 D-55 x-57 }-61 R-65 A-67  -69 v-73
stringa decifrata = F693%B@21/:=<d,C69`%?@2>@2y/6=<9,?69y%E<%,G2m/<9w%B<5,D2x/99}%?<R,A2  /69v

chiave corrente = E
stringa corrente = F693%B@21/:=<d,C69`%?@2>@2y/6=<9,?69y%E<%,G2m/<9w%B<5,D2x/99}%?<R,A2  /69v
stringa decifrata = v96/  2A,R<?%}99/x2D,5<B%w9</m2G,%<E%y96?,9<=6/y2@>2@?%`96C,d<=:/12@B%396F

chiave corrente = B
stringa corrente = v96/  2A,R<?%}99/x2D,5<B%w9</m2G,%<E%y96?,9<=6/y2@>2@?%`96C,d<=:/12@B%396F
stringa decifrata = g-1 s-2 M-3 n-4 l-5 0-6 h-7 a-8  -9 j-10 4-11 m-12-13 Q-14 _-15 %-16 $-17

chiave corrente = C
stringa corrente = g-1 s-2 M-3 n-4 l-5 0-6 h-7 a-8  -9 j-10 4-11 m-12-13 Q-14 _-15 %-16 $-17
stringa decifrata = gsMnl0ha j4mmQ_%$

chiave corrente = A
stringa corrente = gsMnl0ha j4mmQ_%$
stringa decifrata = myStr0ng p4ssW_%$

FINE
```