

Nome: \_\_\_\_\_ Cognome: \_\_\_\_\_ Matr.: \_\_\_\_\_

Architettura degli Elaboratori  
prova scritta di Assembly

ISTRUZIONI:

1. Scrivete subito il vostro nome, cognome e numero di matricola su questo foglio.
1. Aprite un editor a vostra scelta, scrivete una prima linea di commento con le stesse informazioni formattate nel seguente modo: *#matricola nome cognome*  
poi salvate il file chiamandolo: *cognomen.s* (con il vostro cognome e iniziale del nome!).
2. Tutti gli esercizi vanno scritti su questo file.
3. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
4. L'etichetta **main** va messa davanti alla funzione di test dell'esercizio 4 se ci arrivate e funziona altrimenti dovete metterla davanti al primo test (esercizio 2).
5. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
6. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

**Esercizio 1 (2,5 punti)**

Scrivere una funzione **lastIndexOf** che prende come parametri un puntatore ad una stringa e un carattere. La funzione restituisce l'indice dell'ultima occorrenza del carattere nella stringa se c'è altrimenti restituisce -1. Ad esempio, dati: "caramello" e 'a'; la funzione ritorna 3.

**Esercizio 2 (1 punto)**

Scrivere una funzione **lastIndexOfTest** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **lastIndexOf** ai dati e stampa il risultato.

**Esercizio 3 (3,5 punti)**

Scrivere una funzione **setAdd** che prende come parametri un puntatore ad un buffer stringa e una stringa. La funzione aggiunge in coda alla prima stringa tutti i caratteri della seconda che non sono già presenti. La funzione deve richiamare **lastIndexOf**. Ad esempio, dati: "caramello", "dolcetto"; il buffer risulta uguale a: "caramellodt".

**Esercizio 4 (1 punto)**

Scrivere una funzione **setAddTest** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. La funzione chiama **setAdd** e stampa il buffer.

NB. Si ricorda che per stampare un intero lo si deve caricare in `$a0`, caricare 1 in `$v0` e chiamare `syscall`; invece per stampare una stringa la si deve caricare in `$a0`, caricare 4 in `$v0` e chiamare `syscall`.

Nome: \_\_\_\_\_ Cognome: \_\_\_\_\_ Matr.: \_\_\_\_\_

Architettura degli Elaboratori  
prova scritta di Assembly

ISTRUZIONI:

1. Scrivete subito il vostro nome, cognome e numero di matricola su questo foglio.
2. Aprite un editor a vostra scelta, scrivete una prima linea di commento con le stesse informazioni formattate nel seguente modo: *#matricola nome cognome*  
poi salvate il file chiamandolo: *cognomen.s* (con il vostro cognome e iniziale del nome!).
2. Tutti gli esercizi vanno scritti su questo file.
3. Per consegnare, bisogna risolvere i primi due esercizi e verificare con *xspim* che la soluzione sia corretta con i dati dell'esempio.
4. L'etichetta **main** va messa davanti alla funzione di test dell'esercizio 4 se ci arrivate e funziona altrimenti dovete metterla davanti al primo test (esercizio 2).
5. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
6. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

**Esercizio 1 (3 punti)**

Scrivere una funzione **removeDup** che prende come parametri un puntatore ad una stringa e un carattere *c*. La funzione elimina tutte le eventuali occorrenze di *c* dalla stringa ad eccezione della prima. Ad esempio, dati: "barabba" e 'a'; al termine la stringa conterrà: "barbb".

**Esercizio 2 (1 punto)**

Scrivere una funzione **removeDupTest** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **removeDup** ai dati e stampa la stringa.

**Esercizio 3 (3 punti)**

Scrivere una funzione **bag2Set** che prende come parametro un puntatore a stringa. La funzione restituisce la stringa senza caratteri che si ripetono. La funzione deve chiamare la **removeDup**. Ad esempio, dato "scintille"; la funzione restituisce la stringa a: "scintle".

**Esercizio 4 (1 punto)**

Scrivere una funzione **bag2SetTest** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. Il test applica la funzione **bag2Set** ai dati e stampa la stringa.

NB. Si ricorda che per stampare un intero lo si deve caricare in  $\$a0$ , caricare 1 in  $\$v0$  e chiamare `syscall`; invece per stampare una stringa la si deve caricare in  $\$a0$ , caricare 4 in  $\$v0$  e chiamare `syscall`.

Nome: \_\_\_\_\_ Cognome: \_\_\_\_\_ Matr.: \_\_\_\_\_

Architettura degli Elaboratori  
prova scritta di Assembly

ISTRUZIONI:

7. Scrivete subito il vostro nome, cognome e numero di matricola su questo foglio.
3. Aprite un editor a vostra scelta, scrivete una prima linea di commento con le stesse informazioni formattate nel seguente modo: *#matricola nome cognome*  
poi salvate il file chiamandolo: *cognomen.s* (con il vostro cognome e iniziale del nome!).
8. Tutti gli esercizi vanno scritti su questo file.
9. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
10. L'etichetta **main** va messa davanti alla funzione di test dell'esercizio 4 se ci arrivate e funziona altrimenti dovete metterla davanti al primo test (esercizio 2).
11. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
12. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

**Esercizio 1 (3 punti)**

Scrivere una funzione **axn** che prende come parametri tre interi a 8 bit senza segno. La funzione restituisce il primo intero moltiplicato per il secondo moltiplicato per se stesso il terzo volte. Se il terzo è zero restituisce il primo. Ad esempio, dati: 3, 5, 2; la funzione restituisce 75 ( $= 3 * 5^2$ ).

**Esercizio 2 (1 punto)**

Scrivere una funzione **axnTest** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **axn** ai dati e stampa il risultato.

**Esercizio 3 (3 punti)**

Scrivere una funzione **polyEval** che prende come parametro un puntatore ad un array di 4 interi a 8 bit senza segno e un intero  $x$  senza segno. La funzione restituisce la somma dei risultati di **axn** applicata a tutti gli elementi dell'array passandogli l'elemento come primo parametro, la  $x$  come secondo e l'indice dell'elemento come terzo. Ad esempio, dato l'array: 4, 3, 2, 1 e 2; la funzione restituisce: 26 ( $= 4+6+8+8$ ).

**Esercizio 4 (1 punto)**

Scrivere una funzione **polyEvalTest** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. Il test applica la funzione **polyEval** ai dati e stampa il risultato.

NB. Si ricorda che per stampare un intero lo si deve caricare in `$a0`, caricare 1 in `$v0` e chiamare `syscall`; invece per stampare una stringa la si deve caricare in `$a0`, caricare 4 in `$v0` e chiamare `syscall`.

Nome: \_\_\_\_\_ Cognome: \_\_\_\_\_ Matr.: \_\_\_\_\_

Architettura degli Elaboratori  
prova scritta di Assembly

ISTRUZIONI:

7. Scrivete subito il vostro nome, cognome e numero di matricola su questo foglio.
4. Aprite un editor a vostra scelta, scrivete una prima linea di commento con le stesse informazioni formattate nel seguente modo: *#matricola nome cognome*  
poi salvate il file chiamandolo: *cognomen.s* (con il vostro cognome e iniziale del nome!).
8. Tutti gli esercizi vanno scritti su questo file.
9. Per consegnare, bisogna risolvere i primi due esercizi e verificare con *xspim* che la soluzione sia corretta con i dati dell'esempio.
10. L'etichetta **main** va messa davanti alla funzione di test dell'esercizio 4 se ci arrivate e funziona altrimenti dovete metterla davanti al primo test (esercizio 2).
11. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
12. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

**Esercizio 1 (2,5 punti)**

Scrivere una funzione **matchingParen** che prende come parametro un puntatore ad una stringa. La funzione restituisce l'indice della parentesi tonda che chiude la prima che si apre. Si assuma che le parentesi siano bilanciate e ne esista almeno una. Ad esempio, data la stringa: “(())()()”; la funzione restituisce: 9.

**Esercizio 2 (1 punto)**

Scrivere una funzione **matchingParenTest** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **matchingParen** ai dati e stampa il risultato.

**Esercizio 3 (3,5 punti)**

Scrivere una funzione **toSquare** che prende come parametro: un puntatore ad una stringa. La funzione sostituisce le tonde che si aprono in posizione 0 con delle quadre e fa la stessa cosa con il resto della stringa fino alla fine. La funzione deve richiamare **matchingParen**. Ad esempio, dato: “(())()()”; risulta la stringa: “[()()()][()]”.

**Esercizio 4 (1 punto)**

Scrivere una funzione **toSquareTest** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. Il test applica la funzione **toSquare** ai dati e stampa il risultato.

NB. Si ricorda che per stampare un intero lo si deve caricare in `$a0`, caricare 1 in `$v0` e chiamare `syscall`; invece per stampare una stringa la si deve caricare in `$a0`, caricare 4 in `$v0` e chiamare `syscall`.