Matr.:_____ Nome:_____ Cognome: _____

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!).
3. Tutti gli esercizi vanno scritti su questo file.
4. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
5. 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).
6. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
7. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

Esercizio 1 (2 punti)
Scrivere una funzione **`orderedPrefix`** che prende come parametro un puntatore a stringa. La funzione restituisce il numero di caratteri che sono in ordine (ascii) strettamente crescente all'inizio della stringa. Ad esempio, dato: "astro"; la funzione ritorna 3.

Esercizio 2 (2 punti)
Scrivere una funzione **`orderedPrefixTest`** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **`orderedPrefix`** ai dati e stampa il risultato.

Esercizio 3 (3 punti)
Scrivere una funzione **`isOrdered`** che prende come parametro un puntatore a stringa. La funzione restituisce 1 se la stringa è tutta ordinata altrimenti 0. La funzione deve chiamare **`orderedPrefix`**. Ad esempio, dato: "abdefinz"; la funzione restituisce 1.

Esercizio 4 (1 punti)
Scrivere una funzione **`isOrderedTest`** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. La funzione chiama **`isOrdered`** e stampa la stringa se ordinata altrimenti niente.

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`.

Matr.:_____ Nome:_____ Cognome: _____

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!).
3. Tutti gli esercizi vanno scritti su questo file.
4. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
5. 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).
6. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
7. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

Esercizio 1 (3 punti)
Scrivere una funzione **`checkLimit`** che prende tre parametri: un puntatore a un array di interi positivi a 16 bit zero terminato; un intero positivo a 16 bit e un intero *op* che vale 0 oppure 1. Quando *op* vale 0, la funzione restituisce 1 se tutti gli elementi dell'array sono maggiori o uguali all'intero passato; quando *op* vale 1, la funzione restituisce 1 se tutti gli elementi dell'array sono minori o uguali all'intero passato; negli altri casi la funzione restituisce 0. Ad esempio, dati: 3,6,2,14,31,2,0 e 15 e 1; la funzione restituisce: 0.

Esercizio 2 (1 punti)
Scrivere una funzione **`checkLimitTest`** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **`checkLimit`** ai dati e stampa il risultato.

Esercizio 3 (3 punti)
Scrivere una funzione **`checkRange`** che prende tre parametri: un puntatore a un array di interi positivi a 16 bit zero terminato e due interi positivi a 16 bit. La funzione restituisce 1 se tutti i caratteri dell'array sono compresi tra i due interi passati altrimenti restituisce 0. La funzione deve richiamare **`checkLimit`**. Ad esempio, dati: 3,6,2,14,31,2,0 e i due interi 2 e 32; la funzione restituisce: 1.

Esercizio 4 (1 punti)
Scrivere una funzione **`checkRangeTest`** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. La funzione chiama **`checkRange`** 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`.

Matr.:_____ Nome:_____ Cognome: _____

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!).
3. Tutti gli esercizi vanno scritti su questo file.
4. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
5. 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).
6. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
7. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

Esercizio 1 (2 punti)
Scrivere una funzione **`sostituisciOccorrenze`** con tre parametri: un puntatore ad una stringa, un puntatore ad un buffer e un carattere *c*. La funzione copia nel buffer tutti i caratteri della stringa sostituendo le occorrenze di *c* con *c*+1. Il buffer deve essere terminato con zero. Ad esempio, data la stringa: "rastrello" e il carattere 'l'; la funzione scrive nel buffer: "rastremmo".

Esercizio 2 (2 punti)
Scrivere una funzione **`sostituisciOccorrenzeTest`** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **`sostituisciOccorrenze`** ai dati e stampa il buffer.

Esercizio 3 (3 punti)
Scrivere una funzione **`mostraSostituzioni`** che prende come parametri due puntatori a stringa: *testo*, *sostituti*. La funzione per ogni carattere della stringa *sostituti* chiama **`sostituisciOccorrenze`** sulla stringa *testo* e stampa il buffer risultante. Ad esempio, date le stringhe: "rastrello", "lsr"; la funzione stampa: "rastremmo", "rattrello", "sastsello".

Esercizio 4 (1 punti)
Scrivere una funzione **`mostraSostituzioniTest`** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. Il test applica la funzione **`mostraSostituzioni`** ai dati.

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`.

Matr.:_____ Nome:_____ Cognome: _____

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!).
3. Tutti gli esercizi vanno scritti su questo file.
4. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
5. 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).
6. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
7. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

Esercizio 1 (2 punti)
Scrivere una funzione **`sommaOccorrenze`** che prende come parametri un puntatore ad un array di interi a 16 bit senza segno e un intero a 16 bit senza segno. L'array è terminato con uno zero. La funzione restituisce la somma di tutte le occorrenze dell'intero nell'array. Ad esempio, dato l'array: 2,4,4,3,2,3,4,0 e l'intero: 4; la funzione ritorna 12 (4+4+4).

Esercizio 2 (2 punti)
Scrivere una funzione **`sommaOccorrenzeTest`** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **`sommaOccorrenze`** ai dati e stampa il risultato.

Esercizio 3 (3 punti)
Scrivere una funzione **`maxSomma`** che prende come parametro un puntatore ad un array di interi a 16 bit senza segno. L'array è terminato con uno zero. La funzione chiama **`sommaOccorrenze`** passandogli l'array e ciascun intero dell'array stesso e restituisce il risultato massimo. Ad esempio, dato l'array: 2,4,4,3,2,3,4,0; la funzione restituisce 12.

Esercizio 4 (1 punti)
Scrivere una funzione **`maxSommaTest`** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. La funzione chiama **`maxSomma`** 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`.

Matr.:_____ Nome:_____ Cognome: _____

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!).
3. Tutti gli esercizi vanno scritti su questo file.
4. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
5. 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).
6. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
7. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

Esercizio 1 (2 punti)
Scrivere una funzione **`replace`** con tre parametri: un puntatore ad un array (zero terminato) di interi a 8 bit senza segno e due interi ad 8 bit senza segno. La funzione modifica l'array sostituendo tutte le occorrenze del primo intero con il secondo. Ad esempio, dati: 12,3,78,34,12,5,0 e gli interi 12 e 33; la funzione modifica l'array come segue: "33,3,78,34,33,5,0.

Esercizio 2 (2 punti)
Scrivere una funzione **`replaceTest`** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **`replace`** ai dati e stampa l'array modificato.

Esercizio 3 (2 punti)
Scrivere una funzione **`words`** che prende come parametro un puntatore ad un array (zero terminato) di interi a 8 bit senza segno. La funzione sostituisce nell'array gli interi uguali a 32 con degli 0 chiamando **`replace`**. Inoltre la funzione ritorna un puntatore all'intero successivo al primo 32 se c'è altrimenti un puntatore allo zero finale. Ad esempio, dato l'array: 12,7,4,32,45,2,32,7,0; la funzione ritorna un puntatore all'array zero terminato: 45,2,0.

Esercizio 4 (2 punti)
Scrivere una funzione **`wordsTest`** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. Il test applica la funzione **`words`** ai dati e stampa l'array di interi puntato dal 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`.

Matr.:_____ Nome:_____ Cognome: _____

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!).
3. Tutti gli esercizi vanno scritti su questo file.
4. Per consegnare, bisogna risolvere i primi due esercizi e verificare con `xspim` che la soluzione sia corretta con i dati dell'esempio.
5. 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).
6. Chi consegna e ha rispettato quanto scritto sopra prende almeno 3 punti (la sufficienza).
7. In ogni caso, prima di uscire dovete restituirmi questo foglio e dirmi se consegnate oppure no.

Esercizio 1 (3 punti)
Scrivere una funzione **`merge`** che prende come parametri due puntatori a stringhe e un puntatore a un buffer. La funzione assume che le due stringhe contengano due sequenze di caratteri ordinati (ascii) e le copia nel buffer in modo da ottenere una unica stringa ordinata con tutti i caratteri delle due stringhe iniziali. Ad esempio, dati: "bdef" e "acfg"; la funzione scrive nel buffer: "abcdeffg".

Esercizio 2 (1 punti)
Scrivere una funzione **`mergeTest`** senza parametri e con una sezione dati contenente i dati dell'esempio dell'esercizio 1. Il test applica la funzione **`merge`** ai dati e stampa il buffer.

Esercizio 3 (3 punti)
Scrivere una funzione **`isBefore`** che prende come parametri due puntatori a stringhe. La funzione chiama **`merge`** e confronta la prima stringa con la parte iniziale del buffer: se sono uguali restituisce 1 altrimenti 0. Ad esempio, dati: "abc" e "def"; la funzione restituisce 1.

Esercizio 4 (1 punti)
Scrivere una funzione **`isBeforeTest`** senza parametri; con una sezione dati contenente i dati dell'esercizio 3. La funzione chiama **`isBefore`** 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`.