

Fondamenti di Informatica - A.A. 2019-2020

Scuola di Ingegneria Industriale e dell'Informazione
Prof.ssa Cristiana Bolchini
Appello del 20/01/2020



POLITECNICO
MILANO 1863

Cognome

Nome

Matricola o Cod. Persona

| | | | | | | |
|------------------------------|---|---|---|---|---|--------|
| Quesito: | 1 | 2 | 3 | 4 | 5 | Totale |
| Valutazione massima: | 5 | 6 | 5 | 7 | 7 | 30 |
| Valutazione in decimi (/10): | | | | | | |

Istruzioni:

- non è possibile consultare libri, appunti, la calcolatrice o qualsiasi dispositivo elettronico, né comunicare;
- si può scrivere con qualsiasi colore, anche a matita, ad eccezione del rosso.
- tempo a disposizione: 2h

Stile del codice C:

- non è necessario inserire direttive `#include`;
- i commenti non sono necessari, ma potrebbero essere utili nel caso di errore;
- è possibile utilizzare sottoprogrammi di libreria.

Quesito 1 [5 punti]

Dati i due valori $X = -57_{10MS}$ e $Y = 110_{2MS}$ effettuare la conversione in base 2, notazione complemento a 2 ($2C2$), di ognuno degli operandi sul numero **minimo** di bit necessari. Si effettuino quindi le operazioni $X+Y$ e $X-Y$ indicando esplicitamente se si verifica overflow o meno, e motivando la risposta. Mostrare i passaggi fatti.

Riportare qua la codifica di X_{2C2} , Y_{2C2} e i risultati finali delle operazioni, utilizzando solo le caselle necessarie (**allineati a destra**) ed indicando se si è verificato overflow (segnare la casella corrispondente).

X_{2C2} :

Y_{2C2} :

ovf

ovf

$(X + Y)_{2C2}$:

$(X - Y)_{2C2}$:

Quesito 2 [6 punti]

In un array bidimensionale di valori interi, si definisce **dominante** ogni elemento dell'array che è strettamente maggiore di tutti gli elementi dell'array bidimensionale che si trova *in basso a destra* rispetto all'elemento stesso (si veda la figura), non considerando però tutti gli elementi presenti nell'ultima colonna e nell'ultima riga. Si realizzi un sottoprogramma che ricevuto in ingresso un array bidimensionale e qualsiasi altro parametro ritenuto strettamente necessario calcoli e restituisca al chiamante il numero di elementi dominanti presenti. Nel contesto di utilizzo del sottoprogramma, sono presenti le seguenti direttive/istruzioni riportate di seguito.

```
#define NR ...
#define NC ...
...
int main(int argc, char * argv[])
{
    ...
    int info[NR][NC];
}
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5 | 9 | 2 | 4 | 1 | 7 | 2 | 4 |
| 3 | 5 | 6 | 2 | 5 | 6 | 1 | 2 |
| 1 | 3 | 4 | 7 | 8 | 8 | 3 | 0 |
| 1 | 3 | 5 | 6 | 7 | 8 | 2 | 1 |

(a) elemento dominante e array dominato

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 5 | 9 | 2 | 4 | 1 | 7 | 2 | 4 |
| 3 | 5 | 6 | 2 | 5 | 6 | 1 | 2 |
| 1 | 3 | 4 | 7 | 8 | 8 | 3 | 0 |
| 1 | 3 | 5 | 6 | 7 | 8 | 2 | 1 |

(b) elementi dominanti

Quesito 3 [5 punti]

Scrivere un sottoprogramma che riceve in ingresso un array di valori interi v e qualsiasi altro parametro ritenuto necessario, ed altri due valori interi da e a . Il sottoprogramma verifica se nell'array sono presenti tutti e soli i valori inclusi nell'intervallo $[da, a]$, senza ripetizioni. In caso positivo il sottoprogramma restituisce 1, 0 in caso contrario.

Quesito 4 [7 punti]

- (4 punti) Scrivere un sottoprogramma `minmaxstr` che riceve in ingresso una stringa e trasmette al chiamante due caratteri, il minimo e il massimo nell'ordinamento alfabetico tra quelli contenuti nella stringa. Per esempio, se la stringa in ingresso è "esempio", il minimo ed il massimo sono rispettivamente e ed s. La stringa contenga tutti e soli caratteri alfabetici minuscoli.
- (2 punti) Rivedere il sottoprogramma `minmaxstr` in modo tale che i caratteri possano essere sia maiuscoli, sia minuscoli, e facendo in modo che il sottoprogramma trasmetta comunque quelli minuscoli. In tal caso, se la stringa in ingresso è "Architetto", il minimo ed il massimo sono rispettivamente a ed t.
- (1 punto) Scrivere un programma che acquisisce da riga di comando una stringa e chiama il sottoprogramma `minmaxstr`.

Quesito 5 [7 punti]

- (6 punti) Scrivere un sottoprogramma che riceve in ingresso una lista per la gestione dei numeri interi ed un valore intero `dir`. Il sottoprogramma effettua una *rotazione* a sinistra (se `dir` vale 0) o a destra (se `dir` vale 1) del contenuto della lista, restituendola modificata al chiamante. Nella rotazione a sinistra, il primo valore viene posto in fondo alla lista, nello scorrimento a destra, l'ultimo valore della lista viene messo davanti a tutti gli altri. Se per esempio il sottoprogramma riceve in ingresso una lista contenente i valori

$3 \rightarrow 6 \rightarrow 10 \rightarrow -2 \rightarrow 8$

la rotazione a sinistra produce la seguente lista

$6 \rightarrow 10 \rightarrow -2 \rightarrow 8 \rightarrow 3$

la rotazione a destra della lista iniziale produce la seguente lista

$8 \rightarrow 3 \rightarrow 6 \rightarrow 10 \rightarrow -2$

Nel caso in cui la lista è vuota o contiene un solo elemento, il sottoprogramma restituisce la lista ricevuta in ingresso. Si suggerisce di sviluppare due sottoprogrammi separati, uno che esegua la rotazione a sinistra, uno che esegua quella a destra. Non causare *memory leakage*: situazione in cui ci sono nodi non più accessibili senza aver effettivamente rilasciato la memoria.

- (1 punto) Definire un tipo di dato opportuno per gli elementi della lista.

Si considerino già disponibili e non da sviluppare i sottoprogrammi seguenti:

```
/* inserisce in testa alla lista */
elem_t * push(elem_t *, int);
/* inserisce in coda alla lista */
elem_t * append(elem_t *, int);
/* inserisce un elemento nella lista in ordine crescente */
elem_t * insert_inc(elem_t *, int);
/* inserisce un elemento nella lista in ordine decrescente */
elem_t * insert_dec(elem_t *, int);
/* elimina dalla lista il primo elemento */
elem_t * pop(elem_t *);
/* elimina dalla lista tutti gli elementi con il valore indicato */
elem_t * delete(elem_t *, int);
/* restituisce il riferimento all'elemento nella lista che ha il valore indicato, se esiste, NULL altrimenti */
elem_t * exists(elem_t *, int);
/* restituisce il numero di elementi nella lista */
int length(elem_t *);
```

Sapevo già programmare:

☐ No, non è vero ☐ in C ☐ in C++/C# ☐ in Python ☐ in Java ☐ in PHP/Javascript ☐ in VB* ☐ in altro linguaggio