Fondamenti di Informatica A

Esempio di parte teorica di una prova d'esame

Cognome e Nome _____ matr. _____

Domanda 1. Si consideri l'espressione booleana *R* = (*A* **AND** *B*) **XOR** (*A* **OR** *B*). Allora:

- [*V*] [*F*] Se *A* e *B* hanno valore diverso (uno dei due è *true* e l'altro *false*) allora il risultato è sempre *true*
- [*V*] [*F*] Se *A* e *B* hanno lo stesso valore (entrambi *true* o entrambi *false*) allora il risultato è sempre *true*
- [*V*] [*F*] Se *A* e *B* sono entrambi *true*, il risultato è *true*.
- [*V*] [*F*] Se *A* e *B* sono entrambi *false*, il risultato è *true*.

Risposte:

- V
- F: se *A* = *B* = 0, il risultato è 0, se *A* = *B* = 1 il risultato è ancora 0
- F
- F

Domanda 2. Considerando la rappresentazione in complemento a due di interi, utilizzando *N* = 8 bit:

- [*V*] [*F*] Il numero 10010110_{2C} rappresenta un valore positivo
- [*V*] [*F*] La somma di 10010110_{2C} e 00101101_{2C} è un valore negativo
- [*V*] [*F*] 01100011_{2C} rappresenta un numero maggiore di 00110101_{2C}
- [*V*] [*F*] È possibile rappresentare il numero -256

Risposte:

- F
- V: -106 + 45 = -61
- V
- F: il minimo valore rappresentabile in complemento a due con 8 bit è -128

Domanda 3. Un brano audio della durata di 10s viene rappresentato con i parametri seguenti: nessuna forma di compressione; frequenza di campionamento 10kHz (10'000 campioni al secondo); ogni campione è codificato con un valore intero compreso nell'intervallo [-1024, 1023], usando il minimo numero possibile di bit, in complemento a due. Quanti bit sono necessari per memorizzare l'intero file audio?

[Illustrare i calcoli nel riquadro sottostante]

Risposta:

Il numero minimo di bit che sono necessari per rappresentare in complemento a due tutti gli interi nell'intervallo [-1024, 1023] è 11 bit; infatti si ha 10000000000_{2C} = -1024, 01111111111_{2C} = 1023.
Quindi la risposta alla domanda è 10 s $\times$ 10'000 c/s $\times$ 11 bit/c = 1'100'000 bit

Domanda 4. Si consideri la MdT con alfabeto {*blank*, 1}, stato iniziale q0, e tabella delle istruzioni definita come segue:

Stato corrente	Simbolo corrente	Nuovo simbolo	Nuovo stato	Spostamento
q0	1	1	q0	right
q0	*blank*	1	halt	---

Ricordiamo che una *configurazione iniziale* della MdT include il contenuto iniziale del nastro, e la posizione iniziale della testina di lettura-scrittura. Allora:

- [V] [F] Esiste una configurazione iniziale della MdT tale per cui al termine dell'esecuzione il nastro risulti vuoto, ossia tutte le celle siano *blank*.
- [V] [F] Se il nastro contiene inizialmente i simboli 111 e la testina di lettura-scrittura è posizionata sulla prima cifra a sinistra (quella sottolineata), allora al termine dell'esecuzione il nastro conterrà i simboli 1111
- [V] [F] Se il nastro contiene inizialmente i simboli 111 e la testina di lettura-scrittura è posizionata sulla prima cifra a destra (quella sottolineata), allora al termine dell'esecuzione il nastro conterrà i simboli 1111
- [V] [F] Esiste una configurazione iniziale della MdT tale per cui al termine dell'esecuzione sul nastro risulti presente un unico simbolo 1.

Risposte:
- F: se il nastro è inizialmente vuoto, la MdT termina scrivendo un 1; se il nastro è inizialmente non vuoto, nessun simbolo viene cancellato, quindi al termine sarà nuovamente non vuoto
- V
- V
- V: la configurazione iniziale è il nastro vuoto

Domanda 5. Consideriamo il seguente frammento di codice in linguaggio C:

```
int a[] = {10, 7, -1, 4, 2};
int* p = &a[0];
int* q = p+1;
int* r = q+1;
*r = *p
*p = *q;
```

Supponiamo che l'array `a[]` sia memorizzato a partire dall'indirizzo di memoria 1000 (in decimale), e che ogni intero occupi 4 Byte. Allora, al termine dell'esecuzione:

- [V] [F] `p` punta all'indirizzo di memoria 1000
- [V] [F] `r` punta all'indirizzo di memoria 1000
- [V] [F] `a[3]` vale 4
- [V] [F] `a[0]` vale 10

Risposte:
- V
- F: punta a 1008
- V: non viene modificato né direttamente né indirettamente
- F: viene modificato dall'ultima istruzione, vale 7

Domanda 6. Consideriamo la grammatica BNF con simbolo iniziale <A> , simboli terminali “(“ e “)”, e regole di produzione seguenti (omettiamo le virgolette per semplicità):

<A> ::= `(` `)` | `(` <A> `)` | <A> <A>

Allora:

[V] [F] La grammatica può generare la parola `(())()`

[V] [F] La grammatica può generare la parola `((()))`

[V] [F] La grammatica può generare la parola `)((((`

[V] [F] La grammatica può generare la parola `)()()`

Risposte:

- V: in generale, la grammatica genera tutte le sequenze parentesi che sono tra di loro bilanciate
- V
- F
- V