Prova scritta di Algoritmi e Strutture Dati (1^o anno)

Febbraio 2000

NOTE:

- L'ordine in cui vengono svolti gli esercizi non è rilevante (in altre parole: se avete problemi a svolgerne uno potete passare tranquillamente ai successivi e farlo dopo).
- I punti previsti per ogni esercizio si riferiscono ad uno svolgimento completamente corretto.
- Nei calcoli di complessità (esercizi 6 e 7) non fare conti troppo dettagliati esplicitando tutte le costanti, ma non limitarsi neppure a dare solo il risultato: bisogna spiegare come ci si arriva.

NOME:

COGNOME:

MATRICOLA:

==========

Non scrivere qui sotto

Esercizio	Punti previsti	Punti assegnati
1	3	
2	3	
3	2	
4	6	
5	6	
6	5	
7	5	
Totale	30	

Esercizio 1 (punti 3) Consideriamo il seguente codice C:

```
#include <stdio.h>

void p(int *n, int *m)
{ n = m;
  *n = 10;
}

main()
{ int x = 5;
  int y = 1;
  p(&x, &y);
  printf("%d, %d\n", x, y);
}
```

Domanda: Dire qual'è l'output del programma

Esercizio 2 (punti 3)

Consideriamo il seguente codice C:

```
#include <stdio.h>

int a = 1;

void ppp (int * x);

int fff (int x);

int ggg();

main()
{
  int a = 8;
  int b = 100;
  int c;
  ppp(&b);
  c = ggg();
  printf ("%d, %d, %d\n", a, b, c);
}


int ggg()
{
   return(fff(a));
}

void ppp (int * x)
{
   a = 10;
   *x = a + *x;
}

int fff (int x)
{
   return( x + a );
}
```

Domanda: Dire qual'è l'output del programma

Esercizio 3 (punti 2)

Consideriamo le espressioni definite induttivamente come segue:

Lett $= \{A, B, C, \ldots, Z\}$, l'insieme delle lettere maiuscole.

- (base) Lett $\subset$ Exp
- (passo 1) x $\in$ Exp $\Rightarrow$ & x $\in$ Exp
- (passo 2) p, q $\in$ Exp $\Rightarrow$ @ p q $\in$ Exp

Domande:

1. Scrivere una stringa che appartiene ad Exp e contiene almeno 5 lettere e 5 simboli di operazione
2. @ & & @ & A B & C appartiene ad Exp ?

Esercizio 4 (punti 6)

Consideriamo il tipo di dato **Successione** di interi implementato mediante array.

Consideriamo l'operazione

$$\text{DeleteMin} : \text{Succ} \longrightarrow \text{Succ}$$

Definita (a parole) come segue:

DeleteMin(s) elimina da s l'elemento di valore minimo. Se esiste più di un'occorrenza di tale valore, elimina solo la prima. La successione s viene modificata dall'operazione, non se ne fa una copia.

Domande:

1. Definire in pseudo-codice i tipi utilizzati per implementare le successioni.
2. Definire in pseudo-codice l'implementazione di DeleteMin come procedura.

NB: Qualsiasi numero intero è un valore valido! Non si accettano implementazioni in cui si usi un "valore speciale" per marcare caselle vuote o la fine della successione.

Esercizio 5 (punti 6)

Consideriamo il tipo di dato **Dizionario** di interi (senza ripetizioni) implementato mediante alberi binari di ricerca con puntatori figlio sinistro - figlio destro.

Consideriamo l'operazione

$$\text{DeleteMax} : \text{Dizionario} \longrightarrow \text{Dizionario}$$

Definita (a parole) come segue:

DeleteMax(d) elimina da d l'elemento di valore massimo. Il dizionario viene modificato dall'operazione, non se ne fa una copia.

Domande:

1. Definire in pseudo-codice i tipi utilizzati per implementare il dizionario.
2. Definire in pseudo-codice l'implementazione di DeleteMax come procedura.

Esercizio 6 (punti 5)

Consieriamo il pezzo di programma seguente in pseudo-codice:

```
j = 1;
while j < n do
{
      k = 0;
      while k < j do
      {
         aa[k] = aa[k] + aa[k+1];
         k = k+2;
      }
      j = j+3;
}
```

Domanda: determinare la complessità della procedura in $\Theta()$ in funzione di n generico.

Esercizio 7 (punti 5)

Consideriamo la seguente procedura ricorsiva che lavora su alberi binari di interi implementati con puntatori (campo info per il valore del nodo, campi sx e dx per i puntatori ai figli).

```
procedure v(t : albero binario);
{
  if (t non vuoto) then
      { scrivi(t->info);
        if t->info > 0 then
            v(t->sx)
        else
            v(t->dx);
      }
}
```

Domanda: Stimare la complessità della procedura in $\Theta()$ in funzione dell'**altezza** h dell'albero, nel caso peggiore (dire qual'è il caso peggiore).