ANNO ACCADEMICO 2015/16

Algoritmi e Basi di dati – Modulo di Algoritmi e Strutture dati

17 gennaio 2017

1	2	3	4	5
6	7	7	7	6

Esercizio 1

1. Descrivere le seguenti memorizzazioni di un grafo orientato etichettato: a) liste di adiacenza; b) matrici di adiacenza
2. Se il grafo ha n nodi e m archi, calcolare la complessità, per ognuna delle memorizzazioni indicate, dell'operazione di:

i. Inserimento di un arco che congiunge i nodi i e j
ii. Cancellazione di un arco che congiunge i nodi i e j

a-i O(1)
a-ii O(m)
b-i O(1)
b-ii O(1)

Esercizio 2

Calcolare la complessità dell'istruzione

```
for (int i=0; i<=f(t); i++) cout << g(t->left) + g(t->right);
```

in funzione del numero di nodi di t. Indicare per esteso le relazioni di ricorrenza.
Supporre che t sia un albero binario bilanciato e le funzioni **f** e **g** siano definite come segue:

```
int f(Node* t) {
   if (! t)
     return 1;
  return g(t->left) + f(t->left) +
f(t->right);
}
```

```
int g(Node * t) {
  if (!t) return 0;
  int a = g(t->left);
  int b = g(t->right);
  return 1 +2a + 2b;
}
```

Tg(0)= d
Tg(n)= c + 2 Tg(n/2) Tg(n) è O(n)
Rg(0)= d
Rg(n)= c + 4 Rg(n/2) Rg(n) è O(n^2)

Tf(0)= d
Tf(n)= cn + 2 Tf(n/2) Tf(n) è O(nlogn)
Rf(0)= d
Rf(n)= cn^2 + 2 Rf(n/2) Rf(n) è O(n^2)

Numero di iterazioni del for: O(n^2)
Complessità di una iterazione: Tf(n) + 2Tg(n/2)= O(nlogn) + O(n)= O(nlogn)
Complessità del for: O(n^3logn)

Esercizio 3
Sia dato un albero generico ad etichette intere memorizzato figlio-fratello. Si scriva una funzione con complessità O(n) che prende in ingresso un intero positivo k e somma 1 ad ogni foglia di livello maggiore di k.

```
void sum (Node* t, int k, int level) {
   if (! t) return;
   if (level>k && !t->left)t->label++;
   sum level(t->left, k, level+1);
   sum level(t->right, k, level);
}
```

Esercizio 4

a) Algoritmo di Huffmann per la codifica dei caratteri di un alfabeto: dire quale problema risolve, quali sono i suoi input e output, descriverlo dettagliatamente indicando la sua complessità.
b) Fare un esempio con un alfabeto di 4 caratteri per il quale l'algoritmo non presenta miglioramenti rispetto ad una codifica fissa

Esercizio 5

a) Spiegare il meccanismo della derivazione in c++, con riferimento alla visibilità delle variabili e dei metodi delle classi nella gerarchia.
b) Dato il seguente programma, indicarne la gerarchia di classi e l'output indicando riga per riga qual è la funzione coinvolta.

```
class A {
protected:
int x;
public:
A(){ x=10; cout << x << endl;};
void stampa(){ cout << x << endl;};
};
class B: public A {
int x;
public:
B(){ x=20; cout << x << endl;};
void stampa(){ cout << x << endl;};
};
class C: public B {
int x;
public:
C(){ x=30; cout << x << endl;};
void stampa(){ cout << x << endl;};
};
class D: public A {
```

```
public:
D(){ x=50; cout << x << endl;};
void stampa(){ cout << x << endl;};
};

class E: public D {
int x;
public:
E(){ x=40; cout << x << endl;};
void stampa(){ cout << x << endl;};
};

int main(){

A* objA=new A;
B* objB=new B;
C* objC=new C;
D* objD=new D;
E* objE=new E;

B* objH = objC;
objH->stampa();

A* objL=objE;
objL->stampa();
}
```

10	A()
10	A()
20	B()
10	A()
20	B()
30	C()
10	A()
50	D()
10	A()
50	D()
40	E()
20	B.stampa()
50	A.stampa()