Nome e Cognome .... Matricola..

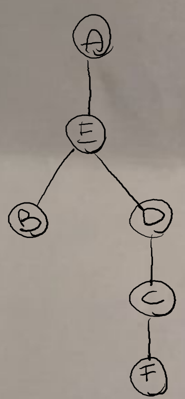
Esame Scritto di Algoritmi 2 del 29/06/2022

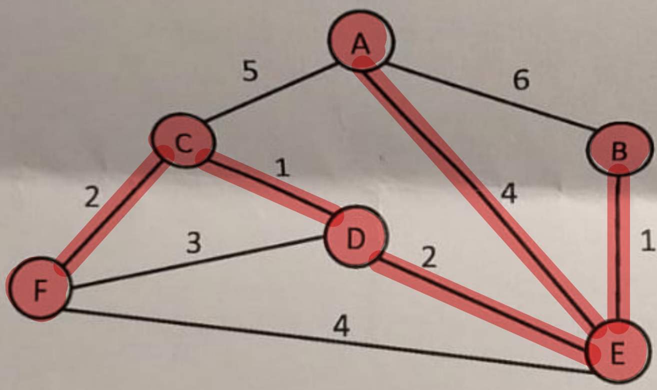
Non è consentito l'uso di libri, appunti e qualsiasi altro materiale. L'esame dura 2 ore.

È possibile consegnare o ritirarsi prima della fine dell'esame (per ritirarsi, scrivere RITIRATO in cima a questo foglio e a quello protocollo). Il punteggio totale è 32 (30 e lode).

ESERCIZIO 1. (Punteggio 7 punti)

1. Si scriva, in pseudocodice, l'algoritmo di Prim per il calcolo del Minimo Albero Ricoprente.
2. Si applichi l'algoritmo al seguente grafo, con vertice di partenza **A** e, dove è possibile effettuare una scelta arbitraria, scegliendo i vertici in ordine alfabetico. Dopo ogni iterazione del ciclo (la riga 0 corrisponde alla situazione iniziale, prima di entrare nel ciclo)
    a. si compili la tabella delle distanze d
    b. si compili la tabella dei vertici ("definitivi") inclusi nella soluzione
    c. si compili la matrice π dei predecessori (disegnandola su foglio o usando d)






```
PRIM
inizializza grafo
C ← inizializza coda di priorità
for ogni v in G
     enqueue(u,C)
     def[v] ← false
while(notEmpty(C)) do
     u ← dequeueMin(C)
     aggiungo u all'albero
     def[u] ← true
     for ogni v adj u
          if def[v]=false and d[v]>W(u,v)
               pred[v] ← u
               d[v] ← W(u,v)
               decreaseKey(C,v,d[v])
```

d	~~A~~	~~B~~	C	~~D~~	~~E~~	F	estratti
0	0_N	∞_N	∞_N	∞_N	∞_N	∞_N	A
1	0_N	6_A	5_A	∞_N	4_A	∞_N	E
2	0_N	1_E	5_A	2_E	4_A	4_E	B
3	0_N	1_E	5_A	2_E	4_A	4_E	D
4	0_N	1_E	1_D	2_E	4_A	3_D	C
5	0_N	1_E	1_D	2_E	4_A	2_C	F
6	0_N	1_E	1_D	2_E	4_A	2_C	

	Vertici inclusi nella soluzione
0	/
1	A
2	A, E
3	A, E, B
4	A, E, B, D
5	A, E, B, D, C
6	A, E, B, D, C, F

```
PQ:{0.03 0.07 0.11 0.15 0.18 0.21 0.25}
PQ:{0.10 0.11 0.15 0.18 0.21 0.25}
PQ:{0.15 0.18 0.21 0.21 0.25}
PQ:{0.21 0.21 0.25 0.33}
PQ:{0.25 0.33 0.42}
PQ:{0.58 0.42}
PQ:{1}
```

ESERCIZIO 2. (Punteggio 7 punti)

1. Dato l'alfabeto composto dai caratteri **a, b, c, d, e, f, g** e la seguente tabella delle frequenze, si calcoli una codifica binaria a lunghezza variabile dell'alfabeto secondo l'algoritmo di Huffman. (Si mostri come la struttura mantenuta dall'algoritmo cambia ad ogni iterazione)

Carattere	a	b	c	d	e	f	g
Frequenza	0.15	0.18	0.07	0.25	0.03	0.11	0.21

2. Quale tecnica algoritmica adotta l'algoritmo di Huffman? (si indichino anche eventuali sottocategorie) Greedy con appetibilità modificabili

ESERCIZIO 3. (Punteggio 6 punti)

1. Utilizzando l'algoritmo visto a lezione, trovare la più lunga sottosequenza comune (LCS) tra le stringhe "LATAKKL" e "AKTAAL". LCS("LATAKKL", "AKTAAL") = ATAL
2. Quale tecnica usa questo algoritmo? (si indichino anche eventuali sottocategorie)
Programmazione dinamica con struttura di memoizzazione (matrice)

ESERCIZIO 4. (Punteggio 8 punti)

Una stringa s si dice palindroma se è uguale letta da sinistra verso destra o viceversa (es. "radar"). Considerata una stringa s, indicizzata come un array, e due indici x e y, la funzione $Pal(x, y, s)$ *restituisce il minimo numero di caratteri necessari da aggiungere per rendere s palindroma, (es. con s = "casa",* $Pal(0,3,\text{"casa"}) = 1$ perché "casaC" è palindroma).

$$Pal(x,y,s) = \begin{cases} 0 & se\ x = y \lor x > y \\ Pal(x+1, y-1, s) & se\ s[x] = s[y] \\ 1 + \min(Pal(x+1,y,s), Pal(x,y-1,s)) & altrimenti \end{cases}$$

Si scriva un algoritmo <u>di programmazione dinamica</u> che, per una qualsiasi stringa s, dati $x = 0$ (indice del primo carattere) e $y = s.length - 1$ (indice dell'ultimo carattere), calcoli $Pal(x, y, s)$ per s. In particolare:

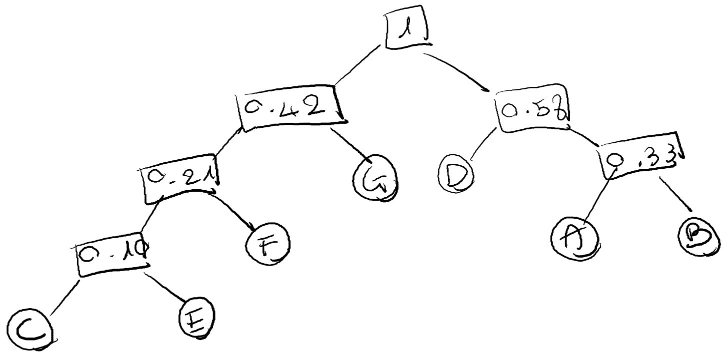
1. Si descriva la struttura dati necessaria per la memoizzazione
2. Si definiscano i casi base, e le loro soluzioni
3. Si dica dove, nella struttura di memoizzazione, sarà presente la soluzione
4. Si scriva in pseudocodice un algoritmo di programmazione dinamica che risolve il problema

ESERCIZIO 5. (Punteggio 4 punti)

a. Il problema dei cammini minimi a sorgente singola può non avere soluzione.
b. Usando la tecnica della ricerca locale, si trovano sempre soluzioni ottime.
c. In un grafo rappresentato con matrice di adiacenza, la rimozione di un vertice ha costo $O(n)$.

a. VERO, nel caso vi siano cicli negativi.
b. FALSO, si trovano solo soluzioni ottime localmente, si tratta comunque di una approssimazione.
c. FALSO, bisogna ricostruire la matrice, e la ricostruzione ha costo O(n^2)

Esercizio 2:



Esercizio 3

		L	A	T	A	K	K	L
A		←	↖	←	↖	←	←	←
K		↑	↑	↑	↑	↖	←	←
T		↑	↑	↖	←	↑	↑	↑
A		↑	↖	↑	↖	←	←	←
A		↑	↖	↑	↖	↑	↑	↑
L		↖	↑	↑	↑	↑	↑	↖

		L	A	T	A	K	K	L
	0	0	0	0	0	0	0	0
A	0	0	1	1	1	1	1	1
K	0	0	1	1	1	2	2	2
T	0	0	1	2	2	2	2	2
A	0	0	1	2	3	3	3	3
A	0	0	1	2	3	3	3	3
L	0	1	1	2	3	3	3	4

Esercizio 4

1. La struttura necessaria per la memoizzazione è una matrice di dimensioni (s+1)x(s+1), dato che sono presenti due parametri (x e y).
2. I casi base sono quelli che non presentano ricorsione: sono quindi x=y o x>y. la loro soluzione è 0 (se la stringa è di lunghezza 0 o 1), altrimenti la parola ricercata, in quanto l'algoritmo conclude le sue chiamate ricorsive.
3. La soluzione sarà presente nella prima cella in alto a destra.
4. PAL(x,y,s)

```
    n = s.length
    P[] ⟵ nuova matrice (s+1)x(s+1)
    for i=0 … n do V[i,0]=0
    for j=0 … n do V[0,j]=0
    for i=1 … n do
        for j=n … 1 do
            if(s[i-1]==s[j-1])
                P[i,j] ⟵ P(i+1,j-1,s)
            else
                P[i,j] ⟵ 1+min(P(i+1,j,s),P(i,j-1,s))
    return P[x,y]
```