Soluzioni seconda prova appello d'esame di Fondamenti di Informatica del 16/12/2020

Esercizi in C

Esercizio 1

Si supponga di aver definito in C una struttura di nome **data** con i campi **giorno**, **mese**, e **anno** di tipo **int** ed una funzione **int giorniTrascorsi(data d1, data d2)** che date due date **d1** e **d2** restituisce il numero di giorni trascorsi tra le due date; la funzione assume che **d1** rappresenti una data precedente **d2**. Scrivere un programma che fa inserire all'utente una sequenza di date; ogni coppia di date consecutive nella seqeunza, che non sono necessariamente in ordine cronologico, definisce un intervallo di tempo; dopo l'inserimento di ogni data il programma mostra all'utente la lunghezza media degli intervalli definiti da due date consecutive nella sequenza. L'inserimento termina quando l'utente inserisce la data 0/0/0, che non fa parte della sequenza.

```
int main(){
      data d1;
      data d2;


      printf("Inserisci giorno mese e anno (0 0 0 per finire)\n");
      scanf("%d%d%d",&d1.giorno,&d1.mese,&d1.anno);

      double somma;
      int count=1;

      while(d1.giorno!=0 && d1.mese!=0 && d1.anno!=0){

            printf("Inserisci giorno mese e anno (0 0 0 per finire)\n");
            scanf("%d%d%d",&d2.giorno,&d2.mese,&d2.anno);

            if(d1.giorno!=0 && d1.mese!=0 && d1.anno!=0){
                  if(d1.anno<d2.anno || (d1.anno==d2.anno &&
                     d1.mese < d2.mese) || (d1.anno==d2.anno &&
                     d1.mese == d2.mese && d1.giorno < d2.giorno)){
                        // d1 precede d2
                        somma+=giorniTrascorsi(d1,d2);
                  }else{
                        // d2 precede d1
                        somma+=giorniTrascorsi(d2,d1);
                  }
                  count++;

                  printf("La media della durata degli intervalli inseriti
                        finora è %g", somma/count);

                  d1.giorno=d2.giorno;
                  d1.mese=d2.mese;
                  d1.anno=d2.anno;
            }
```

```
    }

}
```

Esercizio 2

Si supponga di aver definito in C una struttura di nome **poligonoRegolare** con i campi **numeroLati** di tipo **int**, e **misuraLato** di tipo **double** ed una funzione **double calcolaApotema(poligonoRegolare p)** che dato un poligono regolare **p** restituisce l'apotema di **p**. Scrivere un programma che fa inserire all'utente una sequenza di poligoni specificando per ognuno il numero di lati e la misura del lato e dopo l'inserimento di ogni poligono mostra all'utente l'area media dei poligoni inseriti fino a quel momento. L'inserimento termina quando l'utente inserisce un poligono con 0 lati; tale poligono non fa parte della sequenza.

Si ricorda che l'area di un poligono regolare è pari al prodotto del perimetro per l'apotema diviso due.

```
int main(){
    poligono p;

    printf("Inserisci numero di lati e misura del lato del poligono (0
            lati per finire)\n");
    scanf("%d%lf",&p.numLati,&p.misuraLato);

    double sommaAree=0;
    int count=1;

    while(p.numLati!=0){

        sommaAree=p.numLati*p.misuraLato*calcolaApotema(p)/2;
        printf("La media dell'area dei poligoni inseriti finora è %g",
                somma/count);

        printf("Inserisci numero di lati e misura del lato del
                poligono (0 lati per finire)\n");
        scanf("%d%lf",&p.numLati,&p.misuraLato);
    }
}
```

Esercizio 3

Si supponga di aver definito in C una struttura di nome **poligonoRegolare** con i campi **numeroLati** di tipo **int**, e **misuraLato** di tipo **double** ed una funzione **double numeroFisso(poligonoRegolare p)** che dato un poligono regolare **p** restituisce il numero fisso di **p**. Scrivere un programma che fa inserire all'utente una sequenza di poligoni specificando per ognuno il numero di lati e la misura del lato e dopo l'inserimento di ogni poligono mostra all'utente l'area media dei poligoni inseriti fino a quel momento. L'inserimento termina quando l'utente inserisce un poligono con 0 lati; tale poligono non fa parte della sequenza.

Si ricorda che l'area di un poligono regolare con n lati può essere calcolata conoscendo il lato l e il numero fisso f in base alla seguente formula $A = \frac{n \cdot l^2 \cdot f}{2}$.

```
int main(){
      poligono p;

      printf("Inserisci numero di lati e misura del lato del
            poligono (0 lati per finire)\n");
      scanf("%d%lf",&p.numLati,&p.misuraLato);

      double sommaAree=0;
      int count=1;

      while(p.numLati!=0){


      sommaAree=p.numLati*p.misuraLato*p.misuraLato*numeroFisso(p)/2
;
            printf("La media dell'area dei poligoni inseriti finora è
                  %g", somma/count);

            printf("Inserisci numero di lati e misura del lato del
                  poligono (0 lati per finire)\n");
            scanf("%d%lf",&p.numLati,&p.misuraLato);
      }
}
```

Esercizio 4

Si supponga di aver definito in C una struttura di nome **punto** con i campi **x** e **y** entrambi di tipo **double** ed una funzione **double distanza(punto p1, punto p2)** che dati due punti **p1** e **p2** calcola la loro distanza. Scrivere un programma che fa inserire all'utente una sequenza di punti che rappresentano i vertici di un poligono; l'inserimento termina quando l'utente inserisce un punto che coincide con il primo punto inserito. Al termine dell'inserimento di tutti i vertici del poligono il programma mostra all'utente il valore del perimetro del poligono.

```
int main(){
      punto p0;
      punto p1;
      punto p2;


      printf("Inserisci i punti del poligono (l'ultimo punto deve
                  coincidere con il primo)\n");
      printf("Inserisci un punto\n")
      scanf("%lf%lf",&p0.x,&p0.y);

      p1.x=p0.x;
      p1.y=p0.y;

      printf("Inserisci un punto\n")
```

```
	scanf("%lf%lf",&p2.x,&p2.y);

	while(p2.x!=p0.x && p2.y!=p0.y){

		double perimetro=distanza(p1,p2);

		p1.x=p2.x;
		p1.y=p2.y;


		printf("Inserisci un punto\n")
		scanf("%lf%lf",&p2.x,&p2.y);
	}

	printf("Il perimetro del poligono è: %g ",perimetro);

}
```

Esercizio 5

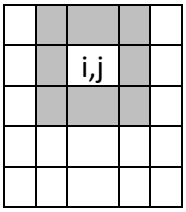
Si supponga di aver definito in C una struttura di nome **orario** con i campi **ore**, **minuti**, e **secondi** di tipo **int** ed una funzione **int secondiTrascorsi(orario o1, orario o2)** che dati due orari **o1** e **o2** restituisce il numero di secondi trascorsi nell'intervallo che intercorre tra i due istanti di tempo **o1** e **o2**; la funzione assume che **o1** rappresenti un istante precedente ad **o2**. Scrivere un programma che fa inserire all'utente una sequenza di orari; ogni coppia di orari consecutivi nella sequenza, che non sono necessariamente in ordine cronologico, definisce un intervallo di tempo; dopo l'inserimento di ogni orario il programma mostra all'utente la lunghezza media degli intervalli definiti da due istanti di tempo successivi nella sequenza. L'inserimento termina quando l'utente inserisce l'orario 0:0:0, che non fa parte della sequenza.

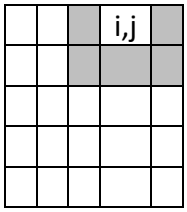
```
int main(){
	orario o1;
	orario o2;


	printf("Inserisci ore minuti e secondi (0 0 0 per finire)\n");
	scanf("%d%d%d",&o1.ore,&o1.minuti,&o1.secondi);

	double somma;
	int count=1;

	while(o1.ore!=0 && o1.minuti!=0 && o1.secondi!=0){

		printf("Inserisci ore minuti e secondi (0 0 0 per finire)\n");
		scanf("%d%d%d",&o2.ore,&o2.minuti,&o2.secondi);

		if(o1.secondi!=0 && o1.minuti!=0 && o1.ore!=0){
			if(o1.ore<o2.ore || (o1.ore==o2.ore &&
			   o1.minuti < o2.minuti) || (o1.ore==o2.ore &&
			   o1.minuti == o2.minuti && o1.secondi < o2.secondi)){
			     // o1 precede o2
			     somma+=secondiTrascorsi(o1,o2);
			}else{
```

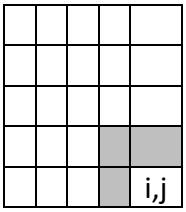
```
                        // o2 precede o1
                        somma+=secondiTrascorsi(o2,o1);
                  }
                  count++;

                  printf("La media della durata degli intervalli inseriti
                        finora è %lf", somma/count);

                  o1.secondi=o2.secondi;
                  o1.minuti=o2.minuti;
                  o1.ore=o2.ore;
            }

      }

}
```

Esercizio 6

L'intorno della cella (**i,j**) è formato da tutte e sole le celle che condividono con la cella (**i,j**) un lato o un vertice (vedi figura). Si osservi che se la cella (**i,j**) non è sul bordo della matrice il suo intorno consiste di 8 celle, se invece essa si trova su uno dei quattro lati della matrice il suo intorno consiste di 5 o 3 celle.La figura mostra alcuni esempi di intorni.





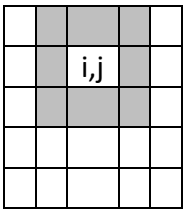


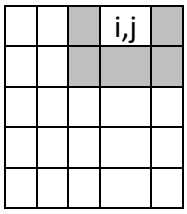
Si supponga di aver definito in C una struttura di nome **coppia** con i campi **a** e **b** di tipo **int**. Si scriva una funzione che riceve in ingresso una matrice di interi, le sue dimensioni e due indici **i** e **j** e restituisce una coppia di interi che rappresentano rispettivamente il numero di elementi nell'intorno della cella (**i,j**) e la somma di tali elementi. Si assuma che gli indici **i** e **j** passati come parametro siano indici validi.

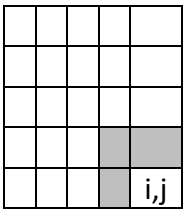
```
coppia analizzaIntorno(int rows, int cols, int mat[][cols], int i, int j){
      coppia res;
      res.a=0;
      res.b=0;

      for(int h=i-1;h<=i+1;h++)
            for(int k=j-1;k<=j+1;k++)
                  if(h>=0 && h<rows && k>=0 && k<cols && h!=i && k!=j){
                        res.a+=mat[h][k];
                        res.b++;
                  }
      return res;
}
```

Esercizio 7

L'intorno della cella (**i,j**) è formato da tutte e sole le celle che condividono con la cella (**i,j**) un lato o un vertice (vedi figura). Si osservi che se la cella (**i,j**) non è sul bordo della matrice il suo intorno consiste di 8 celle, se invece essa si trova su uno dei quattro lati della matrice il suo intorno consiste di 5 o 3 celle.La figura mostra alcuni esempi di intorni.







Si supponga di aver definito in C una struttura di nome **intervallo,** con campi **a** e **b** di tipo **double**, che rappresenta l'intervallo **[a,b]**. Si scriva una funzione che riceve in ingresso una matrice di interi **m**, le sue dimensioni, due indici **i** e **j**, ed un intervallo **r**, e restituisce il numero di elementi nell'intorno della cella (**i,j**) il cui valore è compreso nell'intervallo **r**. Si assuma che gli indici **i** e **j** passati come parametro siano indici validi.

```
int analizzaIntorno(int rows, int cols, int mat[][cols], int i, int j,
      intervallo r){

      int count=0;
      for(int h=i-1;h<=i+1;h++)
            for(int k=j-1;k<=j+1;k++)
                  if(h>=0 && h<rows && k>=0 && k<cols && h!=i && k!=j){
                        if(m[h][k]>=r.a && m[h][k]<=r.b)
                              count++;
                  }
      return count;
}
```

Esercizio 8

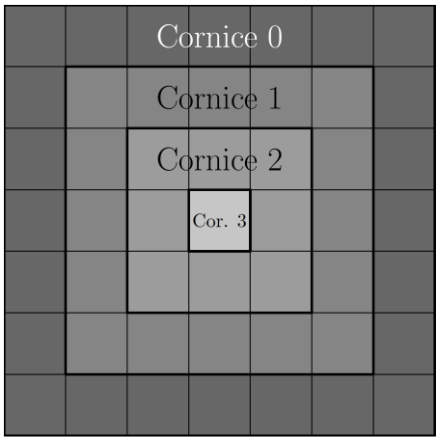
Si supponga di aver definito in C una struttura di nome **coppia** con i campi **a** e **b** di tipo **int**. Si scriva una funzione che riceve in ingresso una matrice di interi **m1**, le sue dimensioni ed una seconda matrice di interi **m2** di dimensione 3 × 3, e verifica se **m2** compare come sottomatrice in **m1**. In caso negativo restituisce la coppia (-1,-1); in caso affermativo restituisce una coppia di interi che rappresentano i due indici **i** e **j** della prima cella della sottomatrice di **m1** che coincide con **m2**.

```
coppia trovaSottomatrice(int rows, int cols, int m1[][cols],
      int m2[][3]){
      coppia res;

      res.a=-1;
      res.b=-1;
      int trovato=0;

      for(int i=0;i<rows && !trovato;i++)
            for(int j=0;j<cols && !trovato;j++){
                  int trovato=1;
```

```
                for(int h=0;h<3;h++)
                      for(int k=0;k<3;k++)
                            if(m1[i+h][j+k]!=m2[i][j])
                                  trovato=0;
                if(trovato){
                      res.a=i;
                      res.b=j;
                }

          }

      return res;
}
```

Esercizio 9

La cornice di indice **k** di una matrice quadrata è l'insieme di celle che distano **k** celle dal bordo. Più precisamente, le celle sul bordo della matrice formano la cornice di indice 0, le celle che si troverebbero sul bordo se si rimuovesse la cornice di indice 0 formano la cornice di indice 1, le celle che si troverebbero sul bordo rimuovendo anche la cornice di indice 1 formano la cornice di indice 2, e così via. Nella figura seguente le cornici sono evidenziate con diverse sfumature di grigio, e per ogni cornice è indicato il corrispondente indice.

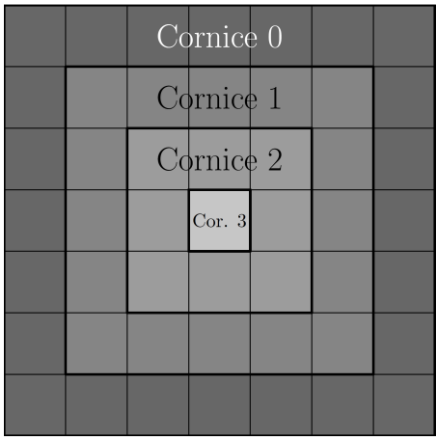


Si supponga di aver definito in C una struttura di nome **coppia** con i campi **a** e **b** di tipo **int**. Si scriva una funzione che riceve in ingresso una matrice quadrata di interi **m**, la sua dimensione ed un indice **k**, e restituisce una coppia di interi che rappresentano rispettivamente il numero di elementi della cornice di indice **k** e la somma di tali elementi. Si assuma che l'indice **k** passato come parametro sia un indice valido.

```
coppia analizzaCornice(int dim, int mat[][dim], int k){
      coppia res;
      res.a=0;
      res.b=0;

      for(int i=0;i<dim;i++)
            for(int j=0;j<dim;j++)
                  if(i==k || i==dim-k-1 || j=k || j=dim-k-1){
                        res.a+=mat[i][j];
                        res.b++;
                  }
      return res;
}
```

Esercizio 10

La cornice di indice **k** di una matrice quadrata è l'insieme di celle che distano **k** celle dal bordo. Più precisamente, le celle sul bordo della matrice formano la cornice di indice 0, le celle che si troverebbero sul bordo se si rimuovesse la cornice di indice 0 formano la cornice di indice 1, le celle che si troverebbero sul bordo rimuovendo anche la cornice di indice 1 formano la cornice di indice 2, e così via. Nella figura seguente le cornici sono evidenziate con diverse sfumature di grigio, e per ogni cornice è indicato il corrispondente indice.



Si supponga di aver definito in C una struttura di nome **intervallo,** con campi **a** e **b** di tipo **double**, che rappresenta l'intervallo **[a,b]**. Si scriva una funzione che riceve in ingresso una matrice quadrata di interi **m**, la sua dimensione, un indice **k**, ed un intervallo **r**, e restituisce il numero di elementi della cornice di indice **k** il cui valore è compreso nell'intervallo **r**. Si assuma che l'indice **k** passato come parametro sia un indice valido.

```
int analizzaCornice(int dim, int mat[][dim], int k, intervallo r){
      int count=0;

      for(int i=0;i<dim;i++)
            for(int j=0;j<dim;j++)
                  if(i==k || i==dim-k-1 || j=k || j=dim-k-1){
                        if(mat[h][k]>=r.a && mat[h][k]<=r.b)
                              count++;
                  }
      return count;
}
```