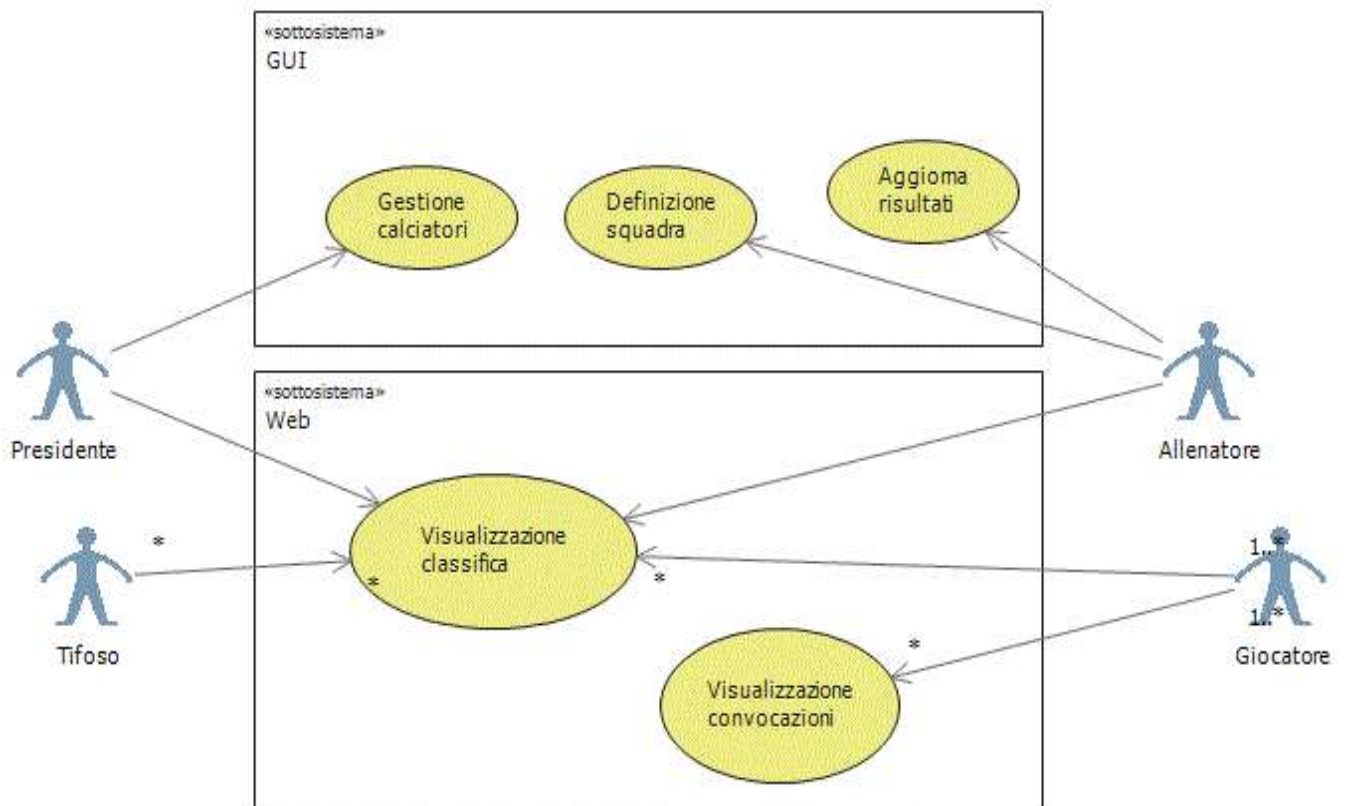


PAOLA PIRRÒ



ESERCIZI DI INFORMATICA 4

PUNTATORI

PUNTATORI N° 1

Caricare e stampare gli elementi di un vettore.

PUNTATORI N° 2

Data una serie di numeri interi, stampare tutti i numeri positivi.

PUNTATORI N° 3

Forniti i voti riportati dalla classe 3 BI, stampare il numero degli studenti che risultano sufficienti.

PUNTATORI N° 4

Un elenco contiene le età di un insieme di persone. Si vogliono visualizzare le età delle persone maggiorenni e la loro posizione nell'elenco.

PUNTATORI N° 5

Caricare e stampare gli elementi di una matrice.

PUNTATORI N° 6

Progettare un'applicazione che permetta di stampare il totale per ogni riga di una matrice, il totale per ogni colonna della matrice e il totale di tutti gli elementi

PUNTATORI N° 7

Si implementi un programma che consenta all'utente di effettuare operazioni di somma e prodotto tra due matrici.

PUNTATORI N° 8

Si sviluppi un sottoprogramma che cerca una stringa in un elenco di stringhe. Il sottoprogramma restituisce un intero che rappresenta vale 1 se la stringa è stata trovata, 0 altrimenti

PUNTATORI N° 9

Si scriva un sottoprogramma che riceve come parametro un array di stringhe e le stampa a schermo in ordine alfabetico.

PUNTATORI N° 10

Sia dato un vettore di caratteri presente in memoria. Si scriva una funzione C che ricevendo in ingresso il vettore e la sua dimensione restituisca, tramite passaggio di parametri, il carattere che più frequentemente degli altri è seguito dal carattere successivo nell'ordine alfabetico. Ad esempio, se il vettore contiene i caratteri A F L M P S T L M la funzione dovrà restituire il carattere L, che per due volte è seguito dal carattere M.

PUNTATORI N° 11

Si dichiarino in C 3 variabili reali (float), di nome x, y, z. Si dichiarino successivamente 3 variabili puntatore, di nome px, py, pz, da inizializzare (direttamente nella dichiarazione), con i valori dei puntatori alle 3 variabili precedenti.

PUNTATORI N° 12

Si realizzi una funzione di nome `scambiaInt`, che scambi il contenuto di due variabili intere. Si faccia un esempio di chiamata della funzione, per scambiare il contenuto delle variabili intere `x` e `y`.

PUNTATORI N° 13

Si scriva una funzione in grado di confrontare due stringhe, ritornando un risultato compatibile con la funzione `strcmp`, salvo il fatto che si devono ignorare le differenze tra caratteri maiuscoli e minuscoli. Si utilizzi una scansione delle stringhe basata su puntatori.

PUNTATORI N° 14

Dopo aver caricato in un vettore di stringhe `n` parole, scrivere un programma che, usando i sottoprogrammi, effettui le seguenti operazioni: cercare una parola scelta dall'utente, stampare l'elenco delle parole ordinate e non.

PUNTATORI N° 15

Dopo aver caricato in un vettore di stringhe `n` parole, scrivere un programma che, usando i sottoprogrammi, effettui le seguenti operazioni: contare il numero di volte che una parola scelta dall'utente, compare nel vettore, stampare l'elenco delle parole, cercare le parole che iniziano con una lettera scelta dall'utente e stamparle.

PUNTATORI N° 16

Si sviluppi un programma che a partire dagli elementi contenuti in due array di interi ordinati costruisce un terzo array ordinato. Per esempio, se i due array di input contengono i seguenti valori: `a1 = {3, 4, 5, 10, 12}`, `a2 = {1, 2, 6, 11, 18}`, l'array risultante conterrà i seguenti valori: `ris = {1, 2, 3, 4, 5, 6, 10, 11, 12, 18}`. Si utilizzino i sottoprogrammi.

ALLOCAZIONE DINAMICA

ALLOCAZIONE DINAMICA N° 1

Si scriva una funzione in grado di creare (e ritornare) una stringa data dalla concatenazione di due stringhe, ricevute come parametri. La stringa generata va allocata dinamicamente.

ALLOCAZIONE DINAMICA N° 2

Si scriva una funzione malloc2d, in grado di allocare una matrice rettangolare di numeri reali (tipo float), le cui dimensioni sono ricevute come parametri. La matrice viene inizializzata azzerando tutte le caselle.

ALLOCAZIONE DINAMICA N° 3

Sia dato un vettore di interi presente in memoria. Si scriva una funzione C che ricevendo il vettore e la sua dimensione restituisca la media degli interi presenti nel vettore non considerando gli eventuali duplicati. Ad esempio, se il vettore contiene gli interi 7 6 4 6 la funzione deve restituire il valore 5.6, ovvero la media di 7, 6 e 4.

ALLOCAZIONE DINAMICA N° 4

Caricare e stampare un vettore la cui dimensione è scelta in modo dinamico,

RECORD

RECORD N° 1

Si implementi un **sottoprogramma** che riceve i seguenti parametri: una struct contenente un array di interi a e un valore intero b di cui si vogliono determinare le occorrenze nel vettore.

RECORD N° 2

Si implementi un sottoprogramma che riceve come parametro le informazioni relative ad un rettangolo ne calcola l'area e restituisce tale valore al chiamante utilizzando il meccanismo del valore di ritorno.

RECORD N° 3

Data la seguente definizione di tipo:

```
typedef struct {int N;int num[10];} numeri;
```

Cosa fa la seguente funzione?

```
int XXX(numeri a)
{int i;
int cont;
cont = 0;
for(i = 0; i < a.N; i++){
if (a.num[i]%2 != 0)
cont++;}
return(cont);}
```

Data la seguente situazione iniziale.

```
a.N = 8;
a.num = {12, 4, 0, 3, 5, 0, 3, 2, 0, 2}
```

Quale valore restituisce la funzione?

LISTE LINEARI

LISTE LINEARI N° 1

Si vuole realizzare un tipo struct, utilizzato per informazioni su operazioni di vendita, avente i seguenti campi:

- ✓ codice: numero intero indicante il codice di riferimento dell'articolo venduto
- ✓ nome: stringa di lunghezza inferiore a 20 caratteri
- ✓ prezzo: numero reale (float) corrispondente al prezzo unitario dell'articolo
- ✓ npezzi: numero (intero) di pezzi venduti

Con tale tipo si vogliono realizzare liste dinamiche. La struct deve quindi essere realizzata come struttura ricorsiva, con puntatore a un dato dello stesso tipo. Si definiscano in C il tipo struct, utilizzando due diversi schemi: (a) una struct a un solo livello, contenente tutti i campi, puntatore ricorsivo compreso; (b) una struct a due livelli, nella quale a primo livello si accede a un puntatore ricorsivo e ad una sotto-struttura contenente il resto dei dati.

LISTE LINEARI N° 2

Si ridefinisca la struct dell'esercizio 1, utilizzando per il campo nome una stringa dinamica. Si scriva poi una funzione in grado di copiare il contenuto di una struttura in un'altra, duplicando la stringa nome. La struttura destinazione viene ricevuta "per indirizzo", mentre la sorgente viene ricevuta "per valore".

LISTE LINEARI N° 3

Si definisca la struttura dati nel caso in cui la pila contenga degli interi e si implementino i sottoprogrammi push e pop.

LISTE LINEARI N° 4

Si definisca una coda che contiene le informazioni relative alle pratiche da evadere nell'ufficio catastale di un comune di piccole dimensioni. Si implementino i sottoprogrammi che operano sulla coda. Infine, si sfruttino questi sottoprogrammi per costruire il programma che consente agli addetti del comune di aggiornare la coda delle pratiche.

TABELLE

TABELLE N° 1

Si vuole scrivere un programma C per la gestione di informazioni relative a studenti neolaureati in ingegneria. Tali informazioni sono contenute in un vettore in cui ogni elemento contiene i seguenti dati così strutturati:

- ✓ Cognome stringa contenuta in campo di esattamente 25 caratteri, può contenere spazi
- ✓ Nome stringa contenuta in campo di esattamente 25 caratteri, può contenere spazi
- ✓ corso di laurea può assumere i seguenti valori: Informatica, Elettronica, Civile, Meccanica, Elettrica, Gestionale
- ✓ voto di laurea intero, indica il voto di laurea
- ✓ lode intero, 1 indica la presenza di lode, 0 indica l'assenza
- ✓ numero anni di iscrizione intero, indica in quanti anni lo studente ha conseguito la laurea

Dopo aver definito una struttura dati (*struct studente*) e il vettore di *struct studente*, definire le seguenti funzioni: *RiempiVettore*, *StudentiMigliori*, *VisualizzaPerCdL*, *MediaCdL*, *main*.

TABELLE N° 2

Si definisca un sottoprogramma che effettua una ricerca per numero di matricola in un array che contiene informazioni relative agli studenti di un corso. Se l'elemento cercato si trova nell' array, il sottoprogramma restituisce al chiamante l'indirizzo della cella corrispondente. Si supponga che l'array su cui si effettua la ricerca sia ordinato.

Si definiscano le strutture dati necessarie per lo sviluppo del sottoprogramma.

TABELLE N° 3

Si definisca:

Un tipo *TipoArchivio* con campi *NumDate* di tipo intero e *SeqDate* array di 100 elementi di tipo *Data*.

Un tipo *VettoreDiPuntatori* come array di 100 elementi di tipo puntatore a elementi di tipo *Data* (*struct giorno*, *mese*, *anno*). Si scriva un *main* che utilizza una variabile *A1* di tipo *TipoArchivio*, una variabile *VP* di tipo *VettoreDiPuntatori* ed una variabile *DataOdierna* di tipo *Data*. Scrivere la porzione di un *main* che, per ogni elemento della Sequenza contenuta nell'archivio *A1*, confronta l'elemento del vettore con la data odierna. Se l'elemento del vettore precede la data odierna il *main* assegna un puntatore del vettore *VP* all'elemento della sequenza appena analizzato. Eseguire l'assegnamento del vettore prima internamente al *main* e poi tramite procedura

TABELLE N° 4

Dopo aver caricato in modo dinamico una tabella di studenti, contenente i relativi nomi, le età e un vettore di voti, stampare la media dei voti di ogni studente con il relativo nome.

FILE DI TESTO

FILE N° 1

Si scriva un programma che legge da un file di testo una sequenza di stringhe e le stampa sullo schermo.

FILE N° 2

Si scriva un programma che acquisisce una sequenza di interi da standard input e la salva su un file in modalità testo.

FILE N° 3

Si scriva un programma che legge da un file di testo una sequenza di interi, la ordina in modo crescente e salva su file la sequenza risultante sovrascrivendo il vecchio contenuto.

FILE N° 4

Si implementi un programma che crea un file in modalità testo e che inserisce nel file valori interi (voti) acquisiti dallo standard input, fino ad un massimo di 20 (voti.txt).

FILE N° 5

Si implementi un programma che legga i dati dal file creato nell'esercizio precedente (voti.txt) e ne calcoli la media stampandola a video ed "appendendola" ad un altro file di testo aperto in modalità "append" (media.txt).

FILE N° 6

Un file contiene una sequenza di stringhe formate da ripetizioni dell'unico carattere '*', separate tra loro da uno o più spazi e ritorni a capo. Le stringhe rappresentano una sequenza di interi positivi codificati in codice unario. Ad esempio, il file

```
*****
** *****
corrisponde alla sequenza 7, 6, 2, 16. Scrivere una funzione C che prende come parametro il nome del file e lo modifica appendendo un fondo al file la media (rappresentata in unario ed arrotondata per difetto) dei valori contenuti nel file. Nell'esempio precedente, all'uscita della funzione il file deve essere il seguente
```

```
*****
***** ** *****
*****
```

in cui il valore 7 (cioè la media arrotondata per difetto dei valori 7, 6, 2 e 16) è stato appeso alla fine del file. Si assuma che ogni riga contenga al più 80 caratteri e, conseguentemente, le stringhe siano composte di al massimo 80 caratteri '*'. Si assuma inoltre che i dati nel file siano corretti, ma che il file di ingresso possa non esistere.

FILE N° 7

Si assuma presente in memoria secondaria un file contenente le informazioni relative alle verbalizzazioni di un esame. Il file contiene le coppie nome-voto tra parentesi; il nome è separato dal voto da uno spazio, le coppie tra parentesi sono separate da spazi o ritorni a capo. Ad esempio, il contenuto del file potrebbe essere il seguente:

(gianni 27) (marco 28) (luigi 20) (giovanni 25) (sergio 24) (luisa 29)

Scrivere un programma che utilizzando una funzione, che ricevendo come parametro il nome del file, stampi la media dei voti ottenuti dagli studenti ed il nome e il voto dello studente col voto più alto. Se più studenti condividono il voto più alto si deve stampare il nome del primo studente tra quelli che hanno il voto più alto.

FILE N° 8

Un file contiene una sequenza (di lunghezza ignota, possibilmente nulla) di valori reali separati da uno spazio. Come esempio si consideri il seguente file

4.522 5.32 4 5.001 16.2 34.2 45.6

Si scriva una funzione C che riceve in input come parametro il nome del file e sostituisce completamente il contenuto del file stesso con i valori interi ottenuti per arrotondamento (all'intero più vicino) dei valori reali presenti nel file. Ad esempio, dopo l'esecuzione della funzione, il contenuto del file dell'esempio deve essere

5 5 4 5 16 34 46.

Se necessario, si utilizzi la funzione `int Arrotonda(float)` che restituisce l'intero più vicino al valore del parametro di tipo `float`. Si assuma che il file sia presente su disco e che la sequenza sia corretta.

FILE N° 9

Cancellare un file il cui nome è ricevuto in input dalla linea di comando.

FILE DATI

FILE N° 1

Si scriva un sottoprogramma che modifica i dati relativi ad un rilievo altimetrico, caratterizzato da ascissa e ordinata (struct punto) e da altezza, memorizzato su file. Il sottoprogramma individua il rilievo da modificare e interagisce con l'utente per ottenere i nuovi dati da inserire nel rilievo.

FILE N° 2

Si sviluppi un programma che consente ad una azienda di mantenere un archivio dei propri impiegati. Il programma verrà utilizzato dal responsabile della gestione del personale dell'azienda oppure da un suo delegato e deve consentire.

- ✓ L'immissione dei dati di un nuovo impiegato.
- ✓ La cancellazione dei dati relativi ad un impiegato che non lavora più per l'azienda in questione.
- ✓ La ricerca dei dati di un impiegato a partire dalla conoscenza del numero di matricola
- ✓ L'aggiornamento dello stipendio di ciascun impiegato. Si suppone che l'aggiornamento venga effettuato.
- ✓ Incrementando della stessa quantità percentuale lo stipendio degli impiegati.
- ✓ La stampa degli stipendi degli impiegati per la gestione delle buste paga.
- ✓ La stampa di tutte le informazioni in archivio.

Le informazioni relative agli impiegati e rilevanti per l'azienda sono il nome e l'indirizzo dell'impiegato, lo stipendio ed il numero di matricola. Lo stipendio viene assegnato all'impiegato al momento della sua assunzione e viene incrementato periodicamente (insieme a quello di tutti gli altri impiegati) utilizzando la funzionalità di aggiornamento degli stipendi. Il programma deve garantire che il numero di matricola sia univoco per ogni impiegato. Infine, il programma deve conservare in modo persistente le informazioni relative agli impiegati.

FILE N° 3

Realizzare un programma che permetta all'utente, tramite un menù, di gestire un archivio di film. Dovranno essere disponibili le seguenti operazioni:

1. L'utente inserisce un certo numero di film:
 - 1.1.1. Codice identificativo numerico e titolo (senza spazi).
 - 1.1.2. I film dovranno essere inseriti con il codice già **ordinato** in senso crescente.
 - 1.1.3. L'inserimento della lista di film è terminato immettendo un codice pari a zero.
2. L'utente inserisce un codice e l'elaboratore verifica se il corrispondente film è presente nel file (ed in questo caso ne scrive i dati sul monitor).

Utilizzare l'algoritmo di ricerca binaria.

OOP

OOP N° 1

Definire una classe convertitore, comprensiva di costruttore e distruttore, al fine di realizzare un convertitore di valute, ad esempio lire/euro. Per il programma si richiede la scrittura di almeno una funzione membro di calcolo, una di output e la definizione di uno o più campi privati ove memorizzare i dati, nonché la scrittura del main.

OOP N° 2

Definire la classe Calcolatrice che possa svolgere le quattro operazioni elementari.

OOP N° 3

Definire la classe Vettore come classe generica che possa caricare, visualizzare e sommare le componenti del vettore.

OOP N° 4

Definire la classe base anagrafica contenente nome, cognome, data di nascita (giorno, mese, anno), sesso con i metodi per inserimento e visualizzazione dei dati. Definire poi la classe derivata studente, contenente anche la classe, la sezione e il campo promosso (tipo bool). Definire anche il main.

OOP N° 5

Definire la classe Rettangolo che permetta di calcolare l'area e il perimetro di un rettangolo ricevendo in input base e altezza. Definire anche il main. Fornire l'output per due rettangoli.

OOP N° 6

Scrivere un programma che utilizzi i metodi degli oggetti string: acquisizione due stringhe, confronto alfabetico, lunghezze, sizeOf, inserimento di caratteri in mezzo a una stringa, ricerca di sottostringhe 'la', estrazioni di sottostringhe, sostituzioni di pezzi di stringhe.

OOP N° 7

Definire la classe frazione che possa ricevere in input numeratore e denominatore, ridurre ai minimi termini la frazione, sommare due frazioni. Definire anche il main.

OOP N° 8

Definire la classe "punto" per rappresentare i punti del piano. Calcolare la distanza del punto dall'origine degli assi. Definire anche il main.

OOP N° 9

Progettare la Calcolatrice che possa svolgere le quattro operazioni elementari.

OOP N° 10

Progettare la Calcolatrice che possa svolgere le quattro operazioni elementari. e implementare l'ereditarietà per estensione.

(sugg: si crea una nuova classe [OperazioniPotenza](#), derivandola da un'esistente [Operazioni](#), che eredita gli attributi e i metodi pubblici della classe [Operazioni](#) e al suo interno si codifica il nuovo metodo che permette l'elevamento a potenza.

OOP N° 11

Progettare la calcolatrice che esegue le quattro operazioni e implementare l'ereditarietà per overriding.

(sugg: si crea una nuova classe *SuperOperazioni*, derivandola da un'esistente *Operazioni*, che eredita gli attributi e i metodi pubblici della classe *Operazioni* e al suo interno si ridefinisce il metodo *Moltiplicazione* come serie di somme successive).

OOP N° 12

Progettare la calcolatrice che esegue le quattro operazioni. (polimorfismo)

(sugg: si progettano due metodi *Moltiplicazione* con lo stesso nome che sono richiamati allo stesso modo ma con parametri di tipo diverso, un metodo lavora con numeri reali e l'altro con numeri interi).

PUNTATORI N° 1

Caricare e stampare gli elementi di un vettore.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

carica

Dati d'input: *vet*, vettore di numeri interi.

Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *i* indice per spazzolare il vettore

Stampa

Dati d'input: *vet*, vettore di numeri interi.

Dati in output: *vet*, vettore di numeri interi

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *i* indice per spazzolare il vettore

main

Dati d'input:.

Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *vet*, vettore di numeri interi

3. Costruzione del modello

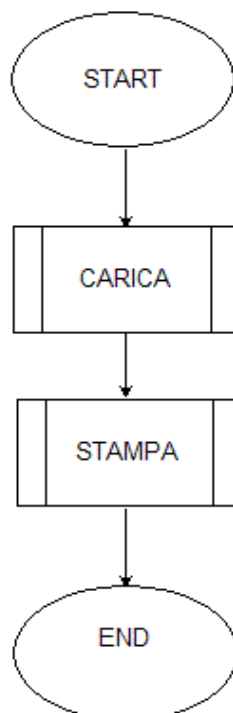
Algoritmo

main

carica

Dati d'input: vet, vettore di numeri

Variabili di lavoro: i, indice che spazzola il vettore



Trace

carica

Passo 1	I	VET[I]	N = 5
I = 1 to N	1		
Leggi		VET[1] = 4	
Next I =5			FALSO

Algoritmo: *stampa*.

Trace: *stampa*.

4. Codifica

BorlandC++ Versione 3.1 Copyright (c) 1990, 1992 Borland International

```

/* Nome del programma: vettore.c
   Programmatore:
   Descrizione: programma che carica e stampa gli elementi di un vettore */
#include <stdio.h>
#include <conio.h>
#define N 5
void carica(int vet[]);
void stampa(int vet[]);
int main(void)
{ int vet[N];
  clrscr();
  carica();
  printf("\n");
  stampa();
  getch();return(0);}
void carica(int vet[])
{ int i;
  for (i = 0; i < N; i++)
    { printf("Inserire l'elemento numero %d ",i);
      scanf("%d",&vet[i]); }}
void stampa(int vet[])
{ int i;
  for (i = 0; i < N; i++)
    printf("%2d\n",vet[i]);
}

```

Microsoft Visual C++

```

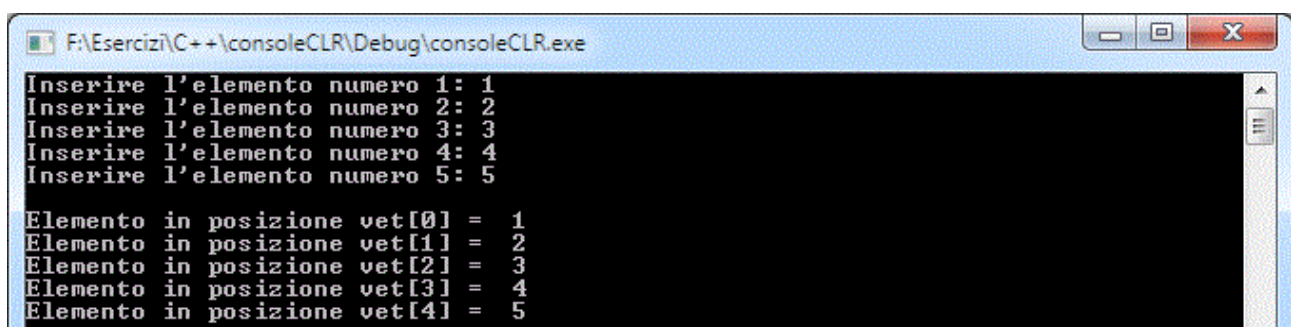
/*****
* Nome del programma: vettore1.c
* Programmatore:
* Descrizione: applicazione che carica e stampa gli elementi di un vettore
*****/
/* direttive del preprocessore*/
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione del vettore */
#define N 5
/* dichiarazioni dei prototipi */
void carica(int *vet);

```

```

void stampa(int *vet);
/* dichiarazioni delle variabili globali: vettore */
int main(void)
{ int vet[N];
  system("cls");
  carica(vet);
  printf("\n");
  stampa(vet);
  system("pause");return(0);}
/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore
Descrizione: limitazioni, errori, modifiche
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void carica(int *vet)
{ /* dichiarazioni: variabili locali */
  for (int i = 0; i < N; ++i) {
    printf("Inserire l'elemento numero %d: ",i+1);
    scanf_s("%d",&vet[i]); }}
/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa gli elementi di un vettore
Descrizione: limitazioni, errori, modifiche;
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void stampa(int *vet)
{ /* dichiarazioni: variabili locali */
  for (int i = 0; i < N; ++i)
    printf("Elemento in posizione vet[%d] = %2d\n",i,vet[i]);
  printf ("\n");}

```



```

F:\Esercizi\C++\consoleCLR\Debug\consoleCLR.exe
Inserire l'elemento numero 1: 1
Inserire l'elemento numero 2: 2
Inserire l'elemento numero 3: 3
Inserire l'elemento numero 4: 4
Inserire l'elemento numero 5: 5

Elemento in posizione vet[0] = 1
Elemento in posizione vet[1] = 2
Elemento in posizione vet[2] = 3
Elemento in posizione vet[3] = 4
Elemento in posizione vet[4] = 5

```

5. Documentazione

6. Testing

PUNTATORI N° 2

Data una serie di numeri interi, stampare tutti i numeri positivi.

1. Identificazione del sistema

Il sistema è di tipo matematico.

2. Analisi dei dati

carica

Dati d'input: *vet*, vettore di numeri interi.

Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *i*, indice che spazzola il vettore.

stampa

Dati d'input: *vet*, vettore di numeri interi.

Dati in output: numeri positivi del vettore

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *i*, indice che spazzola il vettore

main

Dati d'input:.

Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *vet*, vettore di numeri interi

3. Costruzione del modello

4. Codifica

BorlandC++ Versione 3.1 Copyright (c) 1990, 1992 Borland International

```
#include <stdio.h>
#include <conio.h>
#define N 5
void carica(int vet[]);
void stampa(int vet[]);
int main(void)
{ int vet[N];
  clrscr();
  carica(vet);
  printf("\n");
  stampa(vet);
  getch();return(0);}
void carica(int vet[])
{ int i;
  for (i = 0; i < N; i++)
    { printf("Inserire l'elemento numero %d ",i);
      scanf("%d",&vet[i]); }}
void stampa(int vet[])
{ int i;
  for (i = 0; i < N; i++)
    { if (vet[i] > 0) printf("%2d\n",vet[i]); }}
```

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
```



```

#define N 5
/* dichiarazioni dei prototipi */
void carica(int *vet);
void stampa(int *vet);
/* dichiarazioni delle variabili globali: vettore */
int main(void)
{
    int vet[N];
    system("cls");
    carica(vet);
    printf("\n");
    stampa(vet);
    system("pause");return(0);
}
/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore
Descrizione: limitazioni, errori, modifiche
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void carica(int *vet)
{
    int i; /* dichiarazioni: variabili locali */
    for ( i= 0; i < N; ++i) {
        printf("Inserire l'elemento numero %d: ",i+1);
        scanf("%d",&vet[i]);
    }
}
/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa gli elementi di un vettore positivi
Descrizione: limitazioni, errori, modifiche;
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void stampa(int *vet)
{
    int i; /* dichiarazioni: variabili locali */
    for (i = 0; i < N; ++i)
    {
        if (vet[i] > 0)
            printf("Elemento in posizione vet[%d] = %2d\n",i,vet[i]);
        printf ("\n");
    }
}

```

```
C:\Dev-Cpp\puntatori2bis.exe
Inserire l'elemento numero 1: 2
Inserire l'elemento numero 2: 3
Inserire l'elemento numero 3: 4
Inserire l'elemento numero 4: 5
Inserire l'elemento numero 5: 6

Elemento in posizione vet[0] = 2
Elemento in posizione vet[1] = 3
Elemento in posizione vet[2] = 4
Elemento in posizione vet[3] = 5
Elemento in posizione vet[4] = 6
Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
/* *****
* Nome del programma: vettore2.c *
* Programmatore: *
* Descrizione: applicazione che carica e stampa gli elementi positivi di un vettore *
***** */

/* direttive del preprocessore */
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione del vettore */
#define N 5
/* dichiarazioni dei prototipi */
void carica(int *vet);
void stampa(int *vet);
/* dichiarazioni delle variabili globali: vettore */
int main(void)
{ int vet[N];
  system("cls");
  carica(vet);
  printf("\n");
  stampa(vet);
  system("pause");return(0);}

/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore
Descrizione: limitazioni, errori, modifiche
Variabili d'input: vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void carica(int *vet)
{ /* dichiarazioni: variabili locali */
  for (int i = 0; i < N; ++i) {
    printf("Inserire l'elemento numero %d: ",i+1);
    scanf("%d",&vet[i]); }

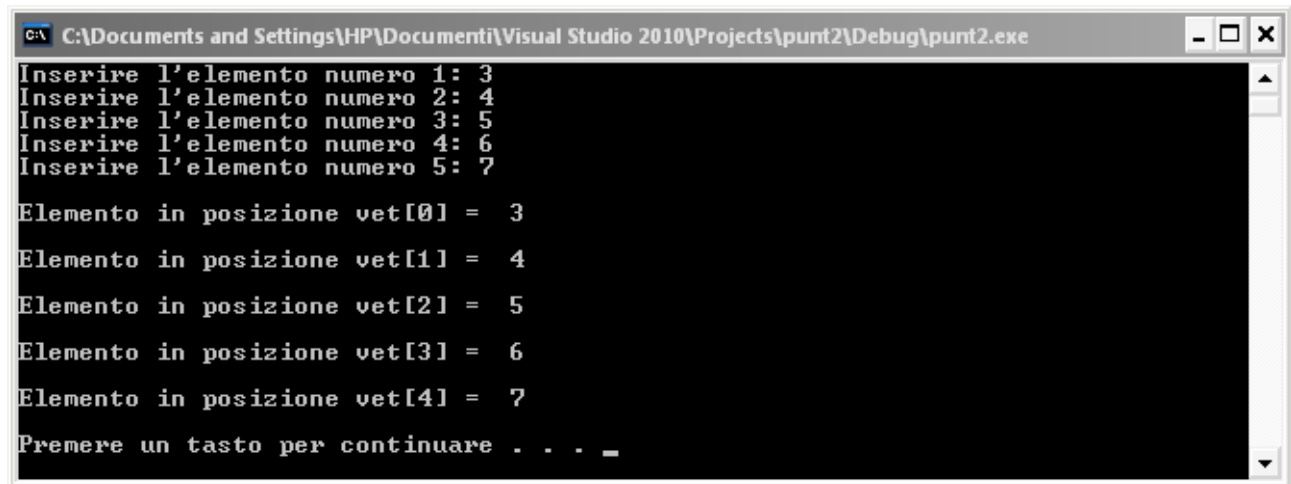
/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa gli elementi di un vettore positivi
Descrizione: limitazioni, errori, modifiche;
```

Variabili d'input: vettore

Variabili di output:

Variabili locali: *i*, variabile di controllo del ciclo for */

```
void stampa(int *vet){  
    for (int i = 0; i < N; ++i)  
        {if (vet[i] > 0)  
            printf("Elemento in posizione vet[%d] = %2d\n", i, vet[i]);  
            printf ("\n");}}
```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt2\Debug\punt2.exe  
Inserire l'elemento numero 1: 3  
Inserire l'elemento numero 2: 4  
Inserire l'elemento numero 3: 5  
Inserire l'elemento numero 4: 6  
Inserire l'elemento numero 5: 7  
  
Elemento in posizione vet[0] = 3  
Elemento in posizione vet[1] = 4  
Elemento in posizione vet[2] = 5  
Elemento in posizione vet[3] = 6  
Elemento in posizione vet[4] = 7  
Premere un tasto per continuare . . . _
```

5. Documentazione

6. Testing

PUNTATORI N° 3

Forniti i voti riportati dalla classe 4 BI, stampare il numero degli studenti che risultano sufficienti.

1. Identificazione del sistema

Il sistema è di tipo matematico.

2. Analisi dei dati

carica

Dati d'input: *vet*, vettore di voti.

Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *i*, indice che spazzola il vettore.

stampa

Dati d'input: *vet*, vettore di numeri interi.

Dati in output: *cont*, contatore delle sufficienze.

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *i*, indice che spazzola il vettore.

main

Dati d'input:.

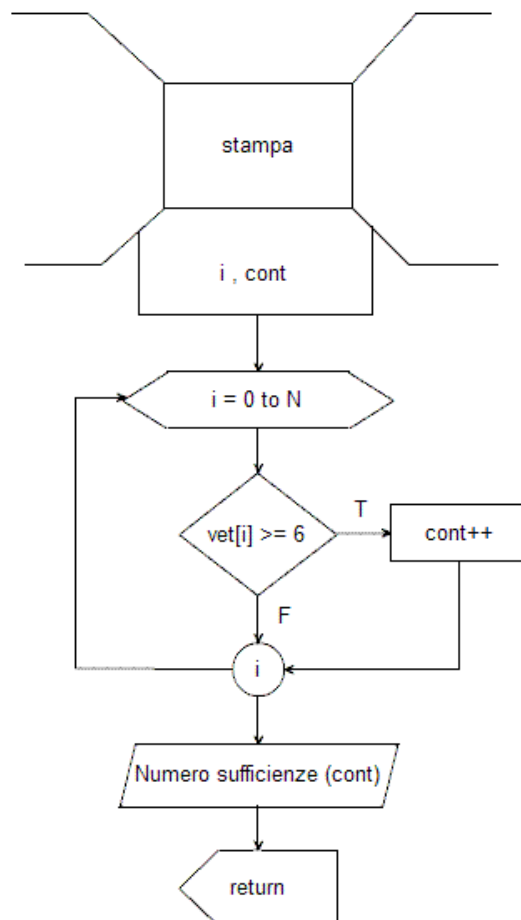
Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *vet*, vettore di numeri interi

3. Costruzione del modello

Algoritmo



4. Codifica

BorlandC++ Versione 3.1 Copyright (c) 1990, 1992 Borland International

```
#include <stdio.h>
#include <conio.h>
#define N 5
void carica(int vet[]);
void stampa(int vet[]);
int main(void)
{ int vet[N];
  clrscr();
  carica(vet);
  printf("\n");
  stampa(vet);
  getch();return(0);}
void carica(int vet[])
{ int i;
  for (i = 0; i < N; i++)
    { printf("Inserire il voto %d ",i);
      scanf("%d",&vet[i]); }}
void stampa(int vet [])
{ int i,cont=0;
  for (i = 0; i < N; i++)
    { if (vet[i] >=6) cont++; }
  printf("Numero di sufficienze della 4 BI %d\n",cont);}
```

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione del vettore */
#define N 5
/* dichiarazioni dei prototipi */
void carica(int *vet);
void stampa(int *vet);
/* dichiarazioni delle variabili globali: vettore */
int main(void)
{ int vet[N];
  system("cls");
  carica(vet);
  printf("\n");
  stampa(vet);
  system("pause");return(0);
}
/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore
Descrizione: limitazioni, errori, modifiche
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
```

```

void carica(int *vet)
{ int i; /* dichiarazioni: variabili locali */
  for ( i = 0; i < N; ++i) {
    printf("Inserire il voto numero %d: ",i+1);
    scanf("%d",&vet[i]);
  }
}
/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa gli elementi di un vettore sufficienti
Descrizione: limitazioni, errori, modifiche;
Variabili d'input:vettore
Variabili di output:
Variabili locali:cont, contatore sufficienti; i, variabile di controllo del ciclo for */
void stampa(int *vet)
{ /* dichiarazioni: variabili locali */
  int i, cont=0;
  for (i = 0; i < N; ++i)
    { if (vet[i] >=6) cont++; }
  printf("Numero di sufficienze della 4 BI  %d\n",cont);
}

```

```

C:\Dev-Cpp\puntatori3.exe
Inserire il voto numero 1: 4
Inserire il voto numero 2: 5
Inserire il voto numero 3: 6
Inserire il voto numero 4: 7
Inserire il voto numero 5: 8

Numero di sufficienze della 4 BI 3
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

/*****
* Nome del programma: vettore3.c
* Programmatore:
* Descrizione: applicazione che carica e stampa i voti sufficienti contenuti in un vettore *
*****/

/* direttive del preprocessore */
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione del vettore */
#define N 5
/* dichiarazioni dei prototipi */
void carica(int *vet);
void stampa(int *vet);
/* dichiarazioni delle variabili globali: vettore */
int main(void)
{ int vet[N];
  system("cls");
  carica(vet);
  printf("\n");
  stampa(vet);
  system("pause");return(0);
}

```

```

/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore
Descrizione: limitazioni, errori, modifiche
Variabili d'input: vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void carica(int *vet)
{ /* dichiarazioni: variabili locali */
    for (int i = 0; i < N; ++i) {
        printf("Inserire il voto numero %d: ", i+1);
        scanf("%d", &vet[i]);
    }
}

/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa gli elementi di un vettore sufficienti
Descrizione: limitazioni, errori, modifiche;
Variabili d'input: vettore
Variabili di output:
Variabili locali: cont, contatore sufficienti; i, variabile di controllo del ciclo for */
void stampa(int *vet)
{ /* dichiarazioni: variabili locali */
    int cont=0;
    for (int i = 0; i < N; ++i)
        { if (vet[i] >=6) cont++; }
    printf("Numero di sufficienze della 4 BI %d\n", cont);
}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt3\Debug\punt3.exe
Inserire il voto numero 1: 4
Inserire il voto numero 2: 3
Inserire il voto numero 3: 5
Inserire il voto numero 4: 6
Inserire il voto numero 5: 7

Numero di sufficienze della 4 BI 2
Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

PUNTATORI N° 4

Un elenco contiene le età di un insieme di persone. Si vogliono visualizzare le età delle persone maggiorenni e la loro posizione nell'elenco.

1. Identificazione del sistema

Il sistema è di tipo matematico.

2. Analisi dei dati

carica

Dati d'input: *vet*, vettore delle età.

Dati in output: *i*, *vet*, indice che spazzola il vettore, cella che contiene l'età.

Costanti: *N*, *MAG*, dimensione del vettore, maggiore età.

Variabili di lavoro: *i*, indice che spazzola il vettore.

stampa

Dati d'input: *vet*, vettore di numeri interi.

Dati in output: età maggiorenni.

Costanti: *N*, *MAG*, dimensione del vettore, maggiore età.

Variabili di lavoro: *i*, indice che spazzola il vettore.

main

Dati d'input:.

Dati in output:

Costanti: *N*, dimensione del vettore.

Variabili di lavoro: *vet*, vettore di numeri interi

3. Costruzione del modello

Algoritmo

Trace

4. Codifica

BorlandC++ Versione 3.1 Copyright (c) 1990, 1992 Borland International

```
#include <stdio.h>
#include <conio.h>
#define N 10
#define MAG 18
void carica(int vet[]);
void stampa(int vet[]);
int main(void)
{ int vet[N];
  clrscr();
  carica(vet);
  printf("\n");
  stampa(vet);
  getch();return(0);
}
void carica(int vet[])
{ int i;
  for (i = 0; i < N; i++)
  { printf("Inserire l'eta' dell'elettore %d ",i);
    scanf("%d",&vet[i]); }
}
void stampa(int vet[])
{ int i;
  for (i = 0; i < N; i++)
```



```

    { if ((vet[i] >= MAG)) printf("Maggiorenne  %d in posizione  %d\n",vet[i],i); }
}

```

Dev-C++ 4.9.9.2

```

#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione del vettore */
#define N 10
#define MAG 18
/* dichiarazioni dei prototipi */
void carica(int *vet);
void stampa(int *vet);
int main(void)
{
    int vet[N];
    system("cls");
    carica(vet);
    printf("\n");
    stampa(vet);
    system("pause");return(0);
}
/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore che sono età
Descrizione: limitazioni, errori, modifiche
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void carica(int *vet)
{
    int i; /* dichiarazioni: variabili locali */
    for ( i = 0; i < N; ++i) {
        printf("Inserire l'elemento numero %d: ",i+1);
        scanf("%d",&vet[i]);
    }
}
/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa le età maggiorenni
Descrizione: limitazioni, errori, modifiche;
Variabili d'input:vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void stampa(int *vet)
{
    int i; /* dichiarazioni: variabili locali */
    for ( i = 0; i < N; ++i)
    {
        if ((vet[i] >= MAG)) printf("Maggiorenne  %d in posizione  %d\n",vet[i],i);
    }
}
}

```

```

C:\Dev-Cpp\puntatori4.exe
Inserire l'elemento numero 1: 34
Inserire l'elemento numero 2: 23
Inserire l'elemento numero 3: 12
Inserire l'elemento numero 4: 34
Inserire l'elemento numero 5: 56
Inserire l'elemento numero 6: 7
Inserire l'elemento numero 7: 8
Inserire l'elemento numero 8: 9
Inserire l'elemento numero 9: 4
Inserire l'elemento numero 10: 3

Maggiorenne 34 in posizione 0
Maggiorenne 23 in posizione 1
Maggiorenne 34 in posizione 3
Maggiorenne 56 in posizione 4
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

/*****
* Nome del programma: vettore4.c
* Programmatore:
* Descrizione: applicazione che carica età e stampa maggiorenni
*****/

/* direttive del preprocessore */
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione del vettore */
#define N 10
#define MAG 18
/* dichiarazioni dei prototipi */
void carica(int *vet);
void stampa(int *vet);
int main(void)
{
    int vet[N];
    system("cls");
    carica(vet);
    printf("\n");
    stampa(vet);
    system("pause"); return(0);
}

/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di un vettore che sono età
Descrizione: limitazioni, errori, modifiche
Variabili d'input: vettore
Variabili di output:
Variabili locali: i, variabile di controllo del ciclo for */
void carica(int *vet)
{
    /* dichiarazioni: variabili locali */
    for (int i = 0; i < N; ++i) {
        printf("Inserire l'elemento numero %d: ", i+1);
        scanf("%d", &vet[i]);
    }
}

/* Nome del sottoprogramma: stampa
Tipo: procedure

```

Scopo: stampa le età maggiorenni

Descrizione: limitazioni, errori, modifiche;

Variabili d'input: vettore

Variabili di output:

Variabili locali: i, variabile di controllo del ciclo for */

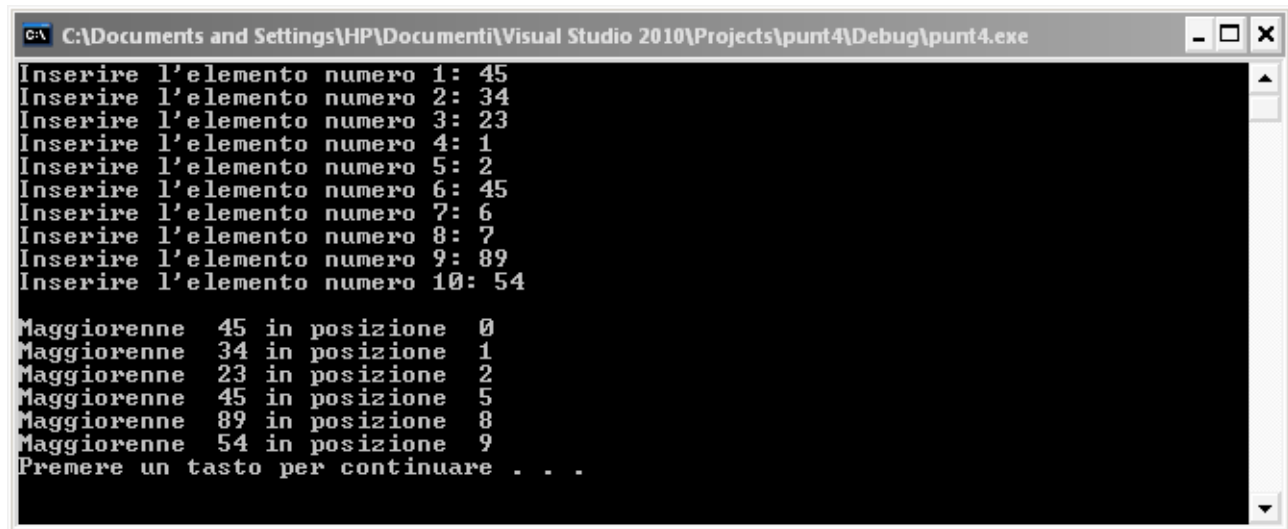
void stampa(int *vet)

{ /* dichiarazioni: variabili locali */

for (int i = 0; i < N; ++i)

{ if ((vet[i] >= MAG)) printf("Maggiorenne %d in posizione %d\n",vet[i],i); }

}



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt4\Debug\punt4.exe
Inserire l'elemento numero 1: 45
Inserire l'elemento numero 2: 34
Inserire l'elemento numero 3: 23
Inserire l'elemento numero 4: 1
Inserire l'elemento numero 5: 2
Inserire l'elemento numero 6: 45
Inserire l'elemento numero 7: 6
Inserire l'elemento numero 8: 7
Inserire l'elemento numero 9: 89
Inserire l'elemento numero 10: 54
Maggiorenne 45 in posizione 0
Maggiorenne 34 in posizione 1
Maggiorenne 23 in posizione 2
Maggiorenne 45 in posizione 5
Maggiorenne 89 in posizione 8
Maggiorenne 54 in posizione 9
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

PUNTATORI N° 5

Caricare e stampare gli elementi di una matrice.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Dati d'input: *mat*, matrice di numeri interi.

Dati in output:

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *i*, *j*, indici che spazzolano la matrice.

stampa

Dati d'input: *mat*, vettore di numeri interi.

Dati in output: *mat*, matrice di numeri interi.

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *i*, *j* indici che spazzolano la matrice.

main

Dati d'input:.

Dati in output:

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *mat*, matrice di numeri interi

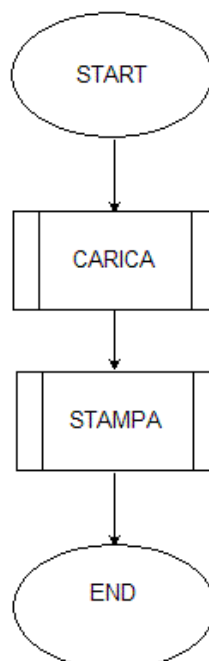
3. Costruzione del modello

Algoritmo

main

carica

Variabili di lavoro: *riga*, *colonna* indici che spazzolano la matrice



Trace: *carica*

Passo 1	riga	colonna	mat[riga,colonna]	C = 4	R = 3
riga = 1 to R	1				
Colonna = 1 to C		1			
Leggi			Mat[1,1] = 4		
Next colonna = 4	2			FALSO	
Next riga = 3					FALSO

4. Codifica

BorlandC++ Versione 3.1 Copyright (c) 1990, 1992 Borland International

/ Nome del programma: matrice.c*

Programmatore:

*Descrizione: programma che carica e stampa gli elementi di una matrice */*

```
#include <stdio.h>
#include <conio.h>
#define R 3
#define C 4
void carica(int mat[R][C]);
void stampa(int mat[R][C]);
int main(void)
{ int mat[R][C];
  clrscr();
  carica(mat);
  printf("\n");
  stampa(mat);
  getch();return(0);
}
void carica(int mat[R][C])
{ int riga, colonna;
  for (riga = 0; riga < R; riga++)
    for (colonna = 0; colonna < C; colonna++)
      { printf("Inserire l'elemento della riga %d e colonna %d ",riga,colonna);
        scanf("%d",&mat[riga][colonna]);
      }
}
void stampa(in mat[R][C])
{ int riga, colonna;
  for (riga = 0; riga < R; riga++)
    { for (colonna = 0; colonna < C; colonna++) { printf("%3d",mat[riga][colonna]); }
      printf("\n");
    }
}
```

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione della matrice */
#define R 3
#define C 4
/* dichiarazioni dei prototipi */
void carica(int mat [R][C]);
void stampa(int mat [R][C]);
/* dichiarazioni delle variabili globali: mat */
```

```

int main(void)
{
    int mat[R][C];
    system("cls");
    carica(mat);
    printf("\n");
    stampa(mat);
    system("pause");return(0);
}
/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di una matrice
Descrizione: limitazioni, errori, modifiche
Variabili d'input:matrice
Variabili di output:
Variabili locali: riga e colonna variabili di controllo del ciclo for */
void carica(int mat [R][C])
{
    int riga, colonna; /* dichiarazioni: variabili locali */
    for ( riga = 0; riga < R; riga++)
        for ( colonna = 0; colonna < C; colonna++) {
            printf("Inserire l'elemento della riga %d e colonna %d: ",riga,colonna);
            scanf("%d",&mat[riga][colonna]);
        }
}
/* Nome del sottoprogramma: stampa
Tipo: procedure
Scopo: stampa gli elementi di una matrice
Descrizione: limitazioni, errori, modifiche;
Variabili d'input:matrice
Variabili di output:
Variabili locali: riga e colonna variabili di controllo del ciclo for */
void stampa(int mat [R][C])
{
    int riga, colonna; /* dichiarazioni: variabili locali */
    for ( riga = 0; riga < R; riga++) {
        for ( colonna = 0; colonna < C; colonna++)
            printf("%4d",mat[riga][colonna]);
        printf("\n");
    }
}

```

```

C:\Documents and Settings\HP\Documenti\puntatori\puntatori5.exe
Inserire l'elemento della riga 0 e colonna 0: 2
Inserire l'elemento della riga 0 e colonna 1: 3
Inserire l'elemento della riga 0 e colonna 2: 4
Inserire l'elemento della riga 0 e colonna 3: 5
Inserire l'elemento della riga 1 e colonna 0: 6
Inserire l'elemento della riga 1 e colonna 1: 7
Inserire l'elemento della riga 1 e colonna 2: 8
Inserire l'elemento della riga 1 e colonna 3: 6
Inserire l'elemento della riga 2 e colonna 0: 1
Inserire l'elemento della riga 2 e colonna 1: 2
Inserire l'elemento della riga 2 e colonna 2: 3
Inserire l'elemento della riga 2 e colonna 3: 4

  2  3  4  5
  6  7  8  6
  1  2  3  4
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

/*****
* Nome del programma: matrice.c
* Programmatore:
* Descrizione:      applicazione che carica e stampa gli elementi di una matrice
*****/

/* direttive del preprocessore */
#include <stdio.h>
#include <stdlib.h>
/* dichiarazioni delle costanti: dimensione della matrice */
#define R 3
#define C 4
/* dichiarazioni dei prototipi */
void carica(int mat [R][C]);
void stampa(int mat [R][C]);
/* dichiarazioni delle variabili globali: mat */
int main(void)
{
    int mat[R][C];
    system("cls");
    carica(mat);
    printf("\n");
    stampa(mat);
    system("pause"); return(0);
}

/* Nome del sottoprogramma: carica
Tipo: procedure
Scopo: carica gli elementi di una matrice
Descrizione: limitazioni, errori, modifiche
Variabili d'input: matrice
Variabili di output:
Variabili locali: riga e colonna variabili di controllo del ciclo for */
void carica(int mat [R][C])
{
    /* dichiarazioni: variabili locali */
    for (int riga = 0; riga < R; riga++)
        for (int colonna = 0; colonna < C; colonna++) {

```

```

        printf("Inserire l'elemento della riga %d e colonna %d: ",riga,colonna);
        scanf("%d",&mat[riga][colonna]);
    }
}
/* Nome del sottoprogramma: stampa
   Tipo: procedure
   Scopo: stampa gli elementi di una matrice
   Descrizione: limitazioni, errori, modifiche;
   Variabili d'input:matrice
   Variabili di output:
   Variabili locali: riga e colonna variabili di controllo del ciclo for */
void stampa(int mat [R][C])
{ /* dichiarazioni: variabili locali */
    for (int riga = 0; riga < R; riga++) {
        for (int colonna = 0; colonna < C; colonna++)
            printf("%4d",mat[riga][colonna]);
        printf("\n");
    }
}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt5\Debug\punt5.exe
Inserire l'elemento della riga 0 e colonna 0: 2
Inserire l'elemento della riga 0 e colonna 1: 2
Inserire l'elemento della riga 0 e colonna 2: 2
Inserire l'elemento della riga 0 e colonna 3: 2
Inserire l'elemento della riga 1 e colonna 0: 1
Inserire l'elemento della riga 1 e colonna 1: 1
Inserire l'elemento della riga 1 e colonna 2: 4
Inserire l'elemento della riga 1 e colonna 3: 5
Inserire l'elemento della riga 2 e colonna 0: 6
Inserire l'elemento della riga 2 e colonna 1: 7
Inserire l'elemento della riga 2 e colonna 2: 8
Inserire l'elemento della riga 2 e colonna 3: 8

    2  2  2  2
    1  1  4  5
    6  7  8  8
Premere un tasto per continuare . . . _

```


PUNTATORI N° 6

Progettare un'applicazione che permetta di stampare il totale per ogni riga di una matrice, il totale per ogni colonna della matrice e il totale di tutti gli elementi

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

carica

Dati d'input: *mat*, matrice di numeri interi.

Dati in output:

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *i*, *j*, indici che spazzolano la matrice.

stampa

Dati d'input: *mat*, matrice di numeri interi.

Dati in output: *mat*, *tot*, matrice di numeri interi, totale di tutti gli elementi della matrice

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *i*, *j*, indici che spazzolano la matrice

totriga

Dati d'input: *mat*, matrice di numeri interi.

Dati in output: *totr*, totali di riga

Costanti: *R*, *C*, numero di righe e colonne della matrice.

Variabili di lavoro: *i*, *j*, indice di riga e colonna per spazzolare la matrice.

totcolonna

Dati d'input: *mat*, matrice di numeri interi.

Dati in output: *totc*, totali di colonna

Costanti: *R*, *C*, numero di righe e colonne della matrice.

Variabili di lavoro: *i*, *j*, indice di riga e colonna per spazzolare la matrice.

main

Dati d'input:.

Dati in output:

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *mat*, matrice di numeri interi

3. Costruzione del modello

Algoritmo

Trace

4. Codifica

BorlandC++ Versione 3.1 Copyright (c) 1990, 1992 Borland International

```
#include <stdio.h>
#include <conio.h>
#define R 3
#define C 3
void carica(int mat [R][C]);
void stampa( int mat [R][C]);
void totriga(int mat [R][C]);
void totcolonna(int mat [R][C]);
int main(void)
{ int mat[R][C];
  clrscr();
```

```

    carica(mat);
    printf("\n");
    totriga(mat); printf("\n");
    totcolonna(mat); printf("\n");
    stampa(mat);
    getch();return(0);
}
void carica(int mat[R][C])
{ int i, j;
  for (i = 0; i < R; i++)
    for (j = 0; j < C; j++)
      { printf("Inserire l'elemento della riga %d e colonna %d ",i,j);
        scanf("%d",&mat[i][j]);
      }
}
void stampa(int mat[R][C])
{ int i, j, tot=0;
  for (i = 0; i < R; i++)
    { for (j = 0; j < C; j++)
      { tot += mat[i][j];
        printf("%2d",mat[i][j]);
      }
      printf("\n");
    }
  printf("\n");
  printf("Il totale di tutti gli elementi della amtrice vale %d\n",tot);
}
void totriga(int mat[R][C])
{ int totr, i, j;
  for (i = 0; i < R; i++)
    { totr = 0;
      for (j = 0; j < C; j++)
        { totr += mat[i][j]; }
      printf("Il totale della riga  %d vale  %d\n",i,totr);
    }
}
void totcolonna(int mat[R][C])
{ int totc, i, j;
  for (j = 0; j < C; j++)
    { totc = 0;
      for (i = 0; i < R; i++)
        { totc += mat[i][j];}
      printf("Il totale della colonna  %d vale  %d\n",j,totc);
    }
}
}

```

Dev-C++ 4.9.9.2

```

#include <stdio.h>
#include <stdlib.h>
#define R 3
#define C 3
void carica(int mat [R][C]);
void stampa( int mat [R][C]);

```

```

void totriga(int mat [R][C]);
void totcolonna(int mat [R][C]);
int main(void)
{ int mat[R][C];
  carica(mat);
  printf("\n");
  totriga(mat); printf("\n");
  totcolonna(mat); printf("\n");
  stampa(mat);
  system("pause");
  return(0);}
void carica(int mat[R][C])
{ int i, j;
  for (i = 0; i < R; i++)
    for (j = 0; j < C; j++)
      { printf("Inserire l'elemento della riga %d e colonna %d ",i,j);
        scanf("%d",&mat[i][j]); }
void stampa(int mat[R][C])
{ int i, j, tot=0;
  for (i = 0; i < R; i++)
    { for (j = 0; j < C; j++)
      { tot += mat[i][j];
        printf("%2d",mat[i][j]); }
      printf("\n"); }
  printf("\n");
  printf("Il totale di tutti gli elementi della amtrice vale %d\n",tot);}
void totriga(int mat[R][C])
{ int totr, i, j;
  for (i = 0; i < R; i++)
    { totr = 0;
      for (j = 0; j < C; j++)
        { totr += mat[i][j]; }
      printf("Il totale della riga %d vale %d\n",i,totr); }}
void totcolonna(int mat[R][C]){
int totc, i, j;
  for (j = 0; j < C; j++) { totc = 0;
    for (i = 0; i < R; i++)
      { totc += mat[i][j];}
    printf("Il totale della colonna %d vale %d\n",j,totc);
  }
}
}

```

```
C:\Documents and Settings\HP\Documenti\puntatori\puntatori6.exe
Inserire l'elemento della riga 0 e colonna 0 3
Inserire l'elemento della riga 0 e colonna 1 4
Inserire l'elemento della riga 0 e colonna 2 5
Inserire l'elemento della riga 1 e colonna 0 6
Inserire l'elemento della riga 1 e colonna 1 7
Inserire l'elemento della riga 1 e colonna 2 8
Inserire l'elemento della riga 2 e colonna 0 9
Inserire l'elemento della riga 2 e colonna 1 9
Inserire l'elemento della riga 2 e colonna 2 7

Il totale della riga 0 vale 12
Il totale della riga 1 vale 21
Il totale della riga 2 vale 25

Il totale della colonna 0 vale 18
Il totale della colonna 1 vale 20
Il totale della colonna 2 vale 20

3 4 5
6 7 8
9 9 7

Il totale di tutti gli elementi della amtrice vale 58
Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
#include <iostream>
#include <conio.h>
using namespace std;
const int R= 3;
const int C= 3;
void carica(int mat[R][C]);
void stampa(int mat[R][C]);
void totriga(int mat[R][C]);
void totcolonna(int mat[R][C]);
int main(void)
{ int mat[R][C];
  cin.clear();
  carica(mat);cout<<"\n";
  totriga(mat); cout<<"\n";
  totcolonna(mat); cout<<"\n";
  stampa(mat);
  getch();return(0);
}
void carica(int mat[R][C])
{ int i, j;
  for (i = 0; i < R; i++)
  for (j = 0; j < C; j++)
  { cout<<"Inserire l'elemento della riga "<<i<<" e colonna "<<j;
    cin>>mat[i][j];}
}
void stampa(int mat[R][C])
{ int i, j, tot=0;
  for (i = 0; i < R; i++)
  { for (j = 0; j < C; j++)
    { tot += mat[i][j];
    cout.width(3);cout<<mat[i][j];}
    cout<<"\n";}
  cout<<"\n";cout<<"Il totale di tutti gli elementi della amtrice vale "<<tot<<"\n";}
void totriga(int mat[R][C])
{ int totr, i, j;
```

```

for (i = 0; i < R; i++)
{ totr = 0;
for (j = 0; j < C; j++)
{ totr += mat[i][j]; }
cout<<"Il totale della riga "<<i<<" vale "<<totr<<"\n";}}
void totcolonna(int mat[R][C])
{ int totc, i, j;
for (j = 0; j < C; j++)
{ totc = 0;
for (i = 0; i < R; i++)
{ totc += mat[i][j];}
cout<<"Il totale della colonna "<<j<<" vale "<<totc<<"\n";
}
}
}

```

```

c:\documents and settings\hp\documenti\visual studio 2010\Projects\punt6\Debug\punt6.exe
Inserire l'elemento della riga 0 e colonna 03
Inserire l'elemento della riga 0 e colonna 14
Inserire l'elemento della riga 0 e colonna 25
Inserire l'elemento della riga 1 e colonna 06
Inserire l'elemento della riga 1 e colonna 17
Inserire l'elemento della riga 1 e colonna 22
Inserire l'elemento della riga 2 e colonna 01
Inserire l'elemento della riga 2 e colonna 13
Inserire l'elemento della riga 2 e colonna 24

Il totale della riga 0 vale 12
Il totale della riga 1 vale 15
Il totale della riga 2 vale 8

Il totale della colonna 0 vale 10
Il totale della colonna 1 vale 14
Il totale della colonna 2 vale 11

  3  4  5
  6  7  2
  1  3  4

Il totale di tutti gli elementi della amtrice vale 35

```

Microsoft Visual C++

```

#include <stdio.h>
#include <stdlib.h>
#define R 3
#define C 3
void carica(int mat [R][C]);
void stampa( int mat [R][C]);
void totriga(int mat [R][C]);
void totcolonna(int mat [R][C]);
int main(void)
{ int mat[R][C];
  carica(mat);
  printf("\n");
  totriga(mat); printf("\n");
  totcolonna(mat); printf("\n");
  stampa(mat);
  system("pause");
  return(0);
}
void carica(int mat[R][C])

```

```

{ int i, j;
  for (i = 0; i < R; i++)
    for (j = 0; j < C; j++)
      { printf("Inserire l'elemento della riga %d e colonna %d ",i,j);
        scanf("%d",&mat[i][j]);
      }
}

void stampa(int mat[R][C])
{ int i, j, tot=0;
  for (i = 0; i < R; i++)
    { for (j = 0; j < C; j++)
      { tot += mat[i][j];
        printf("%2d",mat[i][j]);
      }
      printf("\n");
    }
  printf("\n");
  printf("Il totale di tutti gli elementi della amtrice vale %d\n",tot);
}

void totriga(int mat[R][C])
{ int totr, i, j;
  for (i = 0; i < R; i++)
    { totr = 0;
      for (j = 0; j < C; j++)
        { totr += mat[i][j]; }
      printf("Il totale della riga  %d vale  %d\n",i,totr);
    }
}

void totcolonna(int mat[R][C])
{ int totc, i, j;
  for (j = 0; j < C; j++)
    { totc = 0;
      for (i = 0; i < R; i++)
        { totc += mat[i][j]; }
      printf("Il totale della colonna  %d vale  %d\n",j,totc);
    }
}
}

```

```
c:\documents and settings\hp\documenti\visual studio 2010\Projects\punt6bis\Debug\punt6bis.exe
Inserire l'elemento della riga 0 e colonna 0 2
Inserire l'elemento della riga 0 e colonna 1 3
Inserire l'elemento della riga 0 e colonna 2 4
Inserire l'elemento della riga 1 e colonna 0 5
Inserire l'elemento della riga 1 e colonna 1 6
Inserire l'elemento della riga 1 e colonna 2 7
Inserire l'elemento della riga 2 e colonna 0 8
Inserire l'elemento della riga 2 e colonna 1 9
Inserire l'elemento della riga 2 e colonna 2 10

Il totale della riga 0 vale 9
Il totale della riga 1 vale 18
Il totale della riga 2 vale 27

Il totale della colonna 0 vale 15
Il totale della colonna 1 vale 18
Il totale della colonna 2 vale 21

2 3 4
5 6 7
8 9 10

Il totale di tutti gli elementi della matrice vale 54
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

PUNTATORI N° 7

Si implementi un programma che consenta all'utente di effettuare operazioni di somma e prodotto tra due matrici.

1. Identificazione del sistema

Il sistema è di tipo matematico.

2. Analisi dei dati.

leggiMatrice

Dati d'input: *Matr1*, matrice di numeri interi.

Dati in output:

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *i*, *j*, indici che spazzolano la matrice.

stampa Matrice

Dati d'input: *Matr3*, matrice di numeri interi.

Dati in output: *M*, matrice di numeri interi

Costanti: *R*, *C*, dimensione della matrice.

Variabili di lavoro: *i*, *j*, indici che spazzolano la matrice.

Somma

Dati d'input: *Matr1*, *Matr2*, matrici di cui calcolare la somma, *Matr3* matrice per somma

Dati in output: *Matr3*, matrice per somma

Costanti: *R*, *C*, numero di righe e colonne della matrice.

Variabili di lavoro: *i*, *j*, indice di riga e colonna per spazzolare la matrice.

Prodotto

Dati d'input: *Matr1*, *Matr2*, matrici di cui calcolare il prodotto, *Matr3* matrice per prodotto

Dati in output: *Matr3*, matrice per prodotto

Costanti: *R*, *C*, numero di righe e colonne della matrice.

Variabili di lavoro: *i*, *j*, indice di riga e colonna per spazzolare la matrice.

main

Dati d'input:

Dati in output:

Costanti: *R*, *C*, numero di righe e colonne della matrice.

Variabili di lavoro: *M1*, *M2*, matrici di cui calcolare il prodotto, *M3* matrice per prodotto e somma, *i*, *j*, indice di riga e colonna per spazzolare la matrice.

3. Costruzione del modello

4. Codifica

/* Il seguente programma implementa le funzioni per la lettura, la stampa a video, la somma ed il prodotto di matrici 2x2. Non è stato implementato il sottoprogramma per il calcolo dell'inversa di una matrice */

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define R 2
```

```
#define C 2
```

```
void leggiMatrice(int M[R][C]);
```

```
void stampaMatrice(int M[R][C]);
```

```
void somma(int Matr1[R][C], int Matr2[R][C], int Matr3[R][C]);
```

```
void prodotto(int Matr1[R][C], int Matr2[R][C], int Matr3[R][C]);
```

```
int main(void){
```

```
int M1[R][C], M2[R][C], M3[R][C];
```



```

printf("Inserisci i 4 valori della prima matrice\n");
leggiMatrice(M1);
stampaMatrice(M1);
printf("Inserisci i 4 valori della seconda matrice\n");
leggiMatrice(M2);
stampaMatrice(M2);
somma(M1, M2, M3);
printf("\nSomma di M1 e M2:\n");
stampaMatrice(M3);
prodotto(M1, M2, M3);
printf("\nProdotto di M1 e M2:\n");
stampaMatrice(M3);
system("pause");
return(0);}
/* legge 4 numeri da standard input e li inserisce nella matrice M passata come parametro */
void leggiMatrice(int M[2][2]){
    int i, j;
    for(i=0;i<2;i++)
    for(j=0;j<2;j++)
    scanf("%d",&M[i][j]);}
/* effettua la somma tra le matrici Matr1 e Matr2 e memorizza il risultato in Matr3 */
void somma(int Matr1[2][2], int Matr2[2][2], int Matr3[2][2]){
    int i, j;
    for(i=0;i<2;i++)
    for(j=0;j<2;j++)
    Matr3[i][j]=Matr1[i][j]+Matr2[i][j];}
/* stampa i valori contenuti nella matrice passata come parametro */
void stampaMatrice(int M[2][2]){
    int i, j;
    for(i=0;i<2;i++){
    for(j=0;j<2;j++)
    printf("%d\t",M[i][j]);
    printf("\n");}
    printf("\n\n");}
/* calcola il prodotto tra matrici secondo la formula:
Matr3[i][j] = sommatoria(Matr1[i][k] * Matr2[k][j]).
Matr1 e Matr2 sono le matrici in input e Matr3 viene utilizzata per restituire il risultato dell'operazione. Si noti
la necessità di introdurre tre cicli for innestati. I due cicli più esterni consentono di accedere agli elementi di
Matr3. Il ciclo più interno viene usato per accedere agli elementi appartenenti ad una specifica riga di Matr1
ed ad una specifica colonna di Matr2 */
void prodotto(int Matr1[2][2], int Matr2[2][2], int Matr3[2][2]){
    int i, j, k;
    for (i = 0; i < 2; i++){
    for (j = 0; j < 2; j++){
    Matr3[i][j] = 0;
    for (k = 0; k < 2; k++){
    Matr3[i][j] = Matr3[i][j] + (Matr1[i][k] * Matr2[k][j]);
    }
    }
    }
}

```

```
C:\Dev-Cpp\punt7.exe
4      5
Inserisci i 4 valori della seconda matrice
5
6
7
8      6
5      8
7
Somma di M1 e M2:
7      9
11     13
Prodotto di M1 e M2:
31     36
55     64
Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
void leggiMatrice(int M[2][2]);
void stampaMatrice(int M[2][2]);
void somma(int Matr1[2][2], int Matr2[2][2], int Matr3[2][2]);
void prodotto(int Matr1[2][2], int Matr2[2][2], int Matr3[2][2]);
int main(void){
    int M1[2][2], M2[2][2], M3[2][2];
    printf("Inserisci i 4 valori della prima matrice\n");
    leggiMatrice(M1);
    stampaMatrice(M1);
    printf("Inserisci i 4 valori della seconda matrice\n");
    leggiMatrice(M2);
    stampaMatrice(M2);
    somma(M1, M2, M3);
    printf("Somma di M1 e M2:\n");
    stampaMatrice(M3);
    prodotto(M1, M2, M3);
    printf("Prodotto di M1 e M2:\n");
    stampaMatrice(M3);
    system("pause");
    return(0);}
/* legge 4 numeri da standard input e li inserisce nella matrice M passata come
parametro*/
void leggiMatrice(int M[2][2]){
    int i, j;
    for(i=0; i<2; i++)
        for(j=0; j<2; j++)
            scanf("%d", &M[i][j]);
}
/* effettua la somma tra le matrici Matr1 e Matr2 e memorizza il risultato in Matr3 */
void somma(int Matr1[2][2], int Matr2[2][2], int Matr3[2][2]){
    int i, j;
    for(i=0; i<2; i++)
```

```

for(j=0;j<2;j++)
Matr3[i][j]=Matr1[i][j]+Matr2[i][j];}
/* stampa i valori contenuti nella matrice passata come parametro */
void stampaMatrice(int M[2][2]){
int i, j;
for(i=0;i<2;i++){
for(j=0;j<2;j++)
printf("%d\t",M[i][j]);
printf("\n");}
printf("\n\n");}
/* calcola il prodotto tra matrici secondo la formula:
Matr3[i][j] = sommatoria(Mat1[i][k] * Matr2[k][j]).
Matr1 e Matr2 sono le matrici in input e Matr3 viene utilizzata per restituire il risultato
dell'operazione. Si noti la necessità di introdurre tre cicli for innestati. I due cicli più esterni
consentono di accedere agli elementi di Matr3. Il ciclo più interno viene usato per accedere
agli elementi appartenenti ad una specifica riga di Matr1 ed ad una specifica colonna di
Matr2 */
void prodotto(int Matr1[2][2], int Matr2[2][2], int Matr3[2][2]){
int i, j, k;
for (i = 0; i < 2; i++){
for (j = 0; j < 2; j++){
Matr3[i][j] = 0;
for (k = 0; k < 2; k++){
Matr3[i][j] = Matr3[i][j] + (Matr1[i][k] * Matr2[k][j]);}
}
}
}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt7\Debug\punt7.exe
6
2      4
5      6

Inserisci i 4 valori della seconda matrice
3
4
5
6
3      4
5      6

Somma di M1 e M2:
5      8
10     12

Prodotto di M1 e M2:
26     32
45     56

Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

PUNTATORI N° 8

Si sviluppi un sottoprogramma che cerca una stringa in un elenco di stringhe. Il sottoprogramma restituisce un intero che rappresenta vale 1 se la stringa è stata trovata, 0 altrimenti

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

ricerca

Dati d'input: *vocabolario*, *dim*, *parola*, *M2*, vettore di stringhe, lunghezza massima.*parola* da cercare

Dati in output: 1, se *parola* trovata, 0 se *parola* non trovata

Costanti: *N* dimensione del vettore di stringhe.

Variabili di lavoro: *i*, *trovato*, indice ciclo while, per determinare la presenza o meno della *parola*

main

Dati d'input:

Dati in output:

Costanti: *N* dimensione del vettore di stringhe.

Variabili di lavoro: *i*, *voc*, *p*, indice per spazzolare il vettore di stringhe, *parola* da cercare, vettore di stringhe.

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define N 5
typedef char stringa[N];
int ricerca(stringa vocabolario[], int dim, stringa parola);
int main(void)
{stringa voc[N];
stringa p;
int i;
printf("Inserisci le %d parole del vocabolario\n", N);
for(i=0;i<N;i++)
scanf("%s", voc[i]);
printf("Inserisci la parola da cercare\n");
scanf("%s", p);
if(ricerca(voc, N, p)!=0)
printf("la parola cercata compare\n");
else printf("parola non trovata\n");
system("pause");
return(0);}
int ricerca(stringa vocabolario[], int dim, stringa parola){
int i;
int trovato = 0;
i=0;
while(i<dim && trovato == 0){
if(strcmp(vocabolario[i], parola) == 0)
```

```

trovato = 1;
else i++;}
return trovato;
}

```

```

C:\Dev-Cpp\punt7.exe
Inserisci le 5 parole del vocabolario
casa
mamma
mia
cuore
amore
Inserisci la parola da cercare
amore
la parola cercata compare
Premere un tasto per continuare . . .

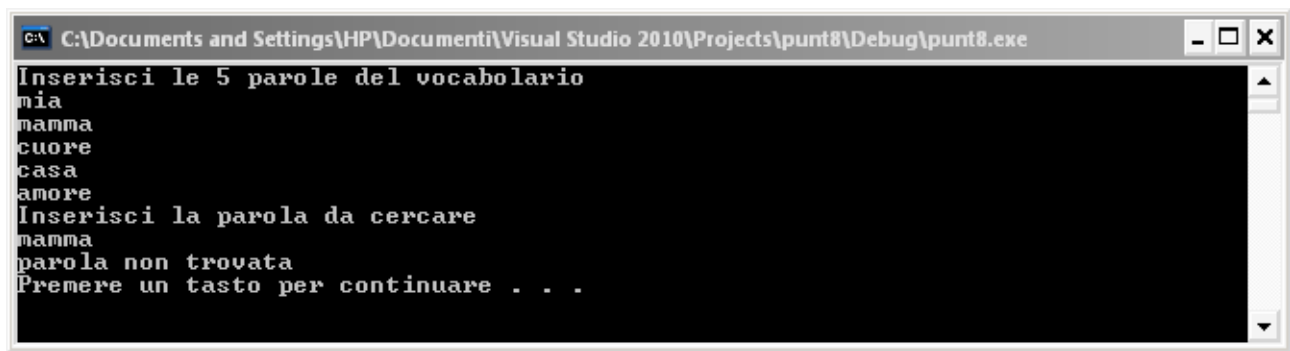
```

Microsoft Visual C++

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define N 5
typedef char stringa[N];
int ricerca(stringa vocabolario[], int dim, stringa parola);
int main(void)
{stringa voc[N];
stringa p;
int i;
printf("Inserisci le %d parole del vocabolario\n", N);
for(i=0;i<N;i++)
scanf("%s", voc[i]);
printf("Inserisci la parola da cercare\n");
scanf("%s", p);
if(ricerca(voc, N, p)!=0)
printf ("la parola cercata compare\n");
else printf ("parola non trovata\n");
system("pause");
return(0);}
int ricerca(stringa vocabolario[], int dim, stringa parola){
int i;
int trovato = 0;
i=0;
while(i<dim && trovato == 0){
if(strcmp(vocabolario[i], parola) == 0)
trovato = 1;
else i++;}
return trovato;
}

```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt8\Debug\punt8.exe
Inserisci le 5 parole del vocabolario
mia
mamma
cuore
casa
amore
Inserisci la parola da cercare
mamma
parola non trovata
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

PUNTATORI N° 9

Si scriva un sottoprogramma che riceve come parametro un array di stringhe e le stampa a schermo in ordine alfabetico.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati.

stampalnOrdine

Dati d'input: *stringhe*, vettore di stringhe

Dati in output: *stringhe*, vettore di stringhe ordinato

Costanti: *N*, *M*, dimensione del vettore di stringhe, dimensione stringa massima.

Variabili di lavoro: *i*, *cont*, *confronto*, *temp*, indici per ordinare il vettore di stringhe, per confrontare le stringhe, per memorizzare la stringa da spostare

main

Dati d'input:

Dati in output:

Costanti: *N*, *M*, dimensione del vettore di stringhe, dimensione stringa massima.

Variabili di lavoro: *stringhe*, vettore di stringhe.

3. Costruzione del modello

Vengono presentate due soluzioni per il problema dato. Nella prima soluzione l'array di stringhe viene prima riordinato alfabeticamente e poi stampato. Nella seconda soluzione, la stampa in ordine alfabetico viene effettuata senza riordinare preventivamente l'array.

4. Codifica

Dev-C++

Prima soluzione

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define N 10
#define M 20
void stampalnOrdine( char stringhe[N][M]);
int main(void){
char stringhe[N][M] =
{"buon\0","anno\0","a\0","tutti\0","gli\0","studenti\0","de\0","corso\0","di\0","informatica!\0"};
stampalnOrdine(stringhe);
system("pause");
return(0);}
/* La funzione stampalnOrdine stampa le stringhe nel vettore passato come parametro in ordine alfabetico.
Riceve due parametri:il numero delle stringhe presenti nel vettore; il vettore delle stringhe */
void stampalnOrdine( char stringhe[N][M])
{int i;
int cont;
/* Per ordinare il vettore si e` utilizzato l'algoritmo dell'Ordinamento Semplice.
Tale algoritmo procede in questo modo:- confronta il primo elemento del vettore con tutti gli altri,se ne trova
uno più piccolo si scambiano le posizioni e questo diventa il nuovo primo elemento. Al termine di questa
prima"passata" il primo elemento e` nella corretta posizione.- confronta il secondo elemento del vettore con
tutti gli altri,se ne trova uno più piccolo si scambiano le posizioni e questo diventa il nuovo secondo
elemento. Al termine di questa seconda"passata" il secondo elemento e` nella corretta posizione.- continua
così fino a che non ha posizionato tutti gli elementi in modo corretto. Da quanto detto si capisce che si deve
procedere con due cicli di for uno dentro l'altro, come descritto nel codice seguente */
/* E' il ciclo più esterno, ad ogni passo l'elemento di posizione "cont" sara` in posizione corretta */
for (cont=0;cont<N;cont++){
/* E' il ciclo più interno. Vengono confrontati l'elemento in posizione "cont" e quello in posizione i-esima. Se
```

```

quest'ultimo è più piccolo viene effettuato lo scambio di posizione*/
for (i=cont; i<N; i++){
int confronto = strcmp(stringhe[i],stringhe[cont]);
if (confronto<0){
/* Scambio fra gli elementi. Viene utilizzata la variabile temporanea temp*/
char temp [M];
strcpy(temp,stringhe[i]);
strcpy(stringhe[i],stringhe[cont]);
strcpy(stringhe[cont],temp);}}}
/* Ciclo di stampa delle stringhe ordinate */
for (i=0; i<N; i++){
printf("%s\n",stringhe[i]);}}

```

```

C:\Dev-Cpp\puntatori9one.exe
a
anno
buon
corso
del
di
gli
informatica!
studenti
tutti
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define N 10
#define M 20
void stampaInOrdine( char stringhe[N][M]);
int main(void){
char stringhe[N][M] =
{"buon\0", "anno\0", "a\0", "tutti\0", "gli\0", "studenti\0", "de\0", "corso\0", "di\0", "informatica!\0"};
stampaInOrdine(stringhe);
system("pause");
return(0);}
/* La funzione stampaInOrdine stampa le stringhe nel vettore passato come parametro in ordine alfabetico.
Riceve due parametri: il numero delle stringhe presenti nel vettore; il vettore delle stringhe */
void stampaInOrdine( char stringhe[N][M])
{int i;
int cont;
/* Per ordinare il vettore si e` utilizzato l'algoritmo dell'Ordinamento Semplice.
Tale algoritmo procede in questo modo:- confronta il primo elemento del vettore con tutti gli altri, se ne trova
uno più piccolo si scambiano le posizioni e questo diventa il nuovo primo elemento. Al termine di questa
prima "passata" il primo elemento e` nella corretta posizione.- confronta il secondo elemento del vettore con
tutti gli altri, se ne trova uno più piccolo si scambiano le posizioni e questo diventa il nuovo secondo elemento.
Al termine di questa seconda "passata" il secondo elemento e` nella corretta posizione.- continua così fino a
che non ha posizionato tutti gli elementi in modo corretto. Da quanto detto si capisce che si deve procedere
con due cicli di for uno dentro l'altro, come descritto nel codice seguente */
/* E' il ciclo più esterno, ad ogni passo l'elemento di posizione "cont" sarà in posizione corretta */
for (cont=0; cont<N; cont++){
/* E' il ciclo più interno. Vengono confrontati l'elemento in posizione "cont" e quello in posizione i-esima. Se
quest'ultimo è più piccolo viene effettuato lo scambio di posizione*/
for (i=cont; i<N; i++){
int confronto = strcmp(stringhe[i],stringhe[cont]);

```



```

if (confronto<0){
/* Scambio fra gli elementi. Viene utilizzata la variabile temporanea temp*/
char temp [M];
strcpy(temp,stringhe[i]);
strcpy(stringhe[i],stringhe[cont]);
strcpy(stringhe[cont],temp);}}}
/* Ciclo di stampa delle stringhe ordinate */
for (i=0; i<N; i++){
printf("%s\n",stringhe[i]);}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt9one\Debug\punt9one.exe
a
anno
buon
corso
del
di
gli
informatica!
studenti
tutti
Premere un tasto per continuare . . .

```

Dev-C++ 4.9.9.2

Seconda soluzione

```

#include <stdio.h>
#include <string.h>
#include<stdlib.h>
#define N 10
#define M 20
void stampaInOrdine( char stringhe[N][M]);
int main(void)
{int i;
char stringhe[N][M] =
{"buon\0","anno\0","a\0","tutti\0","gli\0","studenti\0","de\0","corso\0","di\0","informatica!\0"};
stampaInOrdine(stringhe);
system("pause");
return(0);}
/* La funzione stampaInOrdine stampa le stringhe nel vettore passato come parametro in ordine alfabetico
.Riceve due parametri: il numero delle stringhe presenti nel vettore; il vettore delle stringhe */
void stampaInOrdine( char stringhe[N][M])
{int i;
int cont;
/* questa variabile conterra` l'indice dell'elemento più piccolo del vettore */
int minore;
/* Il sottoprogramma stampando le stringhe in ordine alfabetico .A tal fine procede in questo modo:- cerca
l'elemento più piccolo del vettore e ne memorizza l'indice nella variabile "minore";- stampa l'elemento trovato
e lo sostituisce con la stringa più grande possibile (quella con tutte "z").In questo modo alla prossima ricerca
questa stringa non verrà più considerata perché è la più grande possibile.- ripete i passi precedenti per un
numero di volte pari al numero delle stringhe totali. Da quanto detto si capisce che si deve procedere con
due cicli for uno dentro l'altro, come descritto nel codice seguente */
/* Il ciclo più esterno ripete per N volte la ricerca dell'elemento più piccolo */
for (cont=0;cont<N;cont++){
minore = 0;
/* questo ciclo scorre tutto il vettore per trovare l'elemento più piccolo */
for (i=0; i<N; i++){
/* definizione della variabile che verrà usata per il confronto */
int confronto;
/* E' il ciclo più interno. Vengono confrontati l'elemento in posizione "minore" e quello in posizione i-esima.
Se quest'ultimo e` più piccolo, viene aggiornato il valore di minore */

```

```

confronto = strcmp(stringhe[i],stringhe[minore]);
if (confronto<0){
minore = i;}}
printf("%s\n",stringhe[minore]);
strcpy(stringhe[minore],"zzzzzzzzzzzzzzzzzzzz\0");}}

```

Microsoft Visual C++ seconda soluzione

```

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define N 10
#define M 20
void stampaInOrdine( char stringhe[N][M]);
int main(void)
{int i;
char stringhe[N][M] =
{"buon\0", "anno\0", "a\0", "tutti\0", "gli\0", "studenti\0", "de\0", "corso\0", "di\0", "informatica!\0"};
stampaInOrdine(stringhe);
system("pause");
return(0);}
/* La funzione stampaInOrdine stampa le stringhe nel vettore passato come parametro in ordine alfabetico
.Riceve due parametri: il numero delle stringhe presenti nel vettore; il vettore delle stringhe */
void stampaInOrdine( char stringhe[N][M])
{int i;
int cont;
/* questa variabile conterra l'indice dell'elemento più piccolo del vettore */
int minore;
/* Il sottoprogramma stampando le stringhe in ordine alfabetico .A tal fine procede in questo modo:- cerca
l'elemento più piccolo del vettore e ne memorizza l'indice nella variabile "minore";- stampa l'elemento trovato
e lo sostituisce con la stringa più grande possibile (quella con tutte "z").In questo modo alla prossima ricerca
questa stringa non verrà più considerata perché è la più grande possibile.- ripete i passi precedenti per un
numero di volte pari al numero delle stringhe totali. Da quanto detto si capisce che si deve procedere con due
cicli for uno dentro l'altro, come descritto nel codice seguente */
/* Il ciclo più esterno ripete per N volte la ricerca dell'elemento più piccolo */
for (cont=0;cont<N;cont++){
minore = 0;
/* questo ciclo scorre tutto il vettore per trovare l'elemento più piccolo */
for (i=0; i<N; i++){
/* definizione della variabile che verrà usata per il confronto */
int confronto;
/* E' il ciclo più interno. Vengono confrontati l'elemento in posizione "minore" e quello in posizione i-esima. Se
quest'ultimo e' più piccolo, viene aggiornato il valore di minore */
confronto = strcmp(stringhe[i],stringhe[minore]);
if (confronto<0){
minore = i;}}

```

```
printf("%s\n",stringhe[minore]);  
strcpy(stringhe[minore],"zzzzzzzzzzzzzzzzzzzz\0");}}
```

A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt9two\Debug\punt9two.exe. The window contains the following text:
a
anno
buon
corso
del
di
gli
informatica!
studenti
tutti
Premere un tasto per continuare . . .

5. Documentazione

6. Testing

PUNTATORI N° 10

Sia dato un vettore di caratteri presente in memoria. Si scriva una funzione C che ricevendo in ingresso il vettore e la sua dimensione restituisca, tramite passaggio di parametri, il carattere che più frequentemente degli altri è seguito dal carattere successivo nell'ordine alfabetico. Ad esempio, se il vettore contiene i caratteri A F L M P S T L M la funzione dovrà restituire il carattere L, che per due volte è seguito dal carattere M.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Conta

Dati d'input: *v, n, puntatore a una stringa, numero caratteri della stringa

Dati in output: max, che più frequentemente degli altri è seguito dal carattere successivo nell'ordine alfabetico

Variabili di lavoro: i, app, indice per spazzolare la stringa, vettore in cui memorizza per ogni lettera il numero di volta che è seguita della sua successiva.

3. Costruzione del modello

La funzione conta() utilizza un vettore app in cui memorizza per ogni lettera il numero di volta che è seguita della sua successiva. Successivamente, la funzione cerca il massimo del vettore app. Il vettore è allocato di dimensione pare al numero di lettere dell'alfabeto (cioè 26), indicato tramite l'espressione 'Z' - 'A' + 1.

4. Codifica

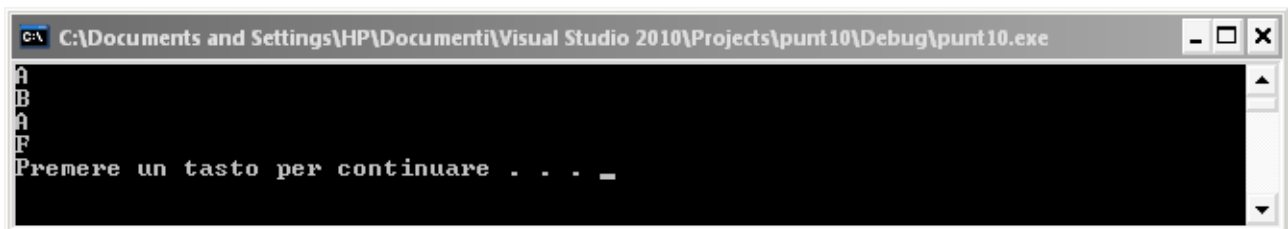
Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include<stdlib.h>
char conta(char *v, int n );
int main(void){
printf("%c\n",conta("ABABC",5));
printf("%c\n",conta("BCFLZ",5));
printf("%c\n",conta("ABCDE",5));
printf("%c\n",conta("FFGFF",5));
system("pause");
return(0);}
char conta(char *v, int n ){
int i, app['Z'-'A'+1]; /* assumo i caratteri tutti maiuscoli */
int max = 0;
for (i = 0; i < ('Z'-'A'+1); ++i)
app[i] = 0;
for (i = 0 ; i < n-1; ++i)
if (v[i+1] == v[i] + 1)
++app[v[i]-'A'];
for (i = 1; i < ('Z'-'A'+1); ++i)
if (app[i] > app[max])
max = i;
return( max+'A');
}
```



Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
char conta(char *v, int n );
int main(void){
    printf("%c\n", conta("ABABC", 5));
    printf("%c\n", conta("BCFLZ", 5));
    printf("%c\n", conta("ABCDE", 5));
    printf("%c\n", conta("FFGFF", 5));
    system("pause");
    return(0);}
char conta(char *v, int n ){
    int i, app['Z'-'A'+1]; /* assumo i caratteri tutti maiuscoli */
    int max = 0;
    for (i = 0; i < ('Z'-'A'+1); ++i)
        app[i] = 0;
    for (i = 0 ; i < n-1; ++i)
        if (v[i+1] == v[i] + 1)
            ++app[v[i]-'A'];
    for (i = 1; i < ('Z'-'A'+1); ++i)
        if (app[i] > app[max])
            max = i;
    return( max+'A');}
```



5. Documentazione

6. Testing

PUNTATORI N° 11

Si dichiarino in C 3 variabili reali (float), di nome x, y, z. Si dichiarino successivamente 3 variabili puntatore, di nome px, py, pz, da inizializzare (direttamente nella dichiarazione), con i valori dei puntatori alle 3 variabili precedenti.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello

4. Codifica

```
int x, y, z;  
int *px=&x, py=&y, pz=&z;
```

5. Documentazione

6. Testing

PUNTATORI N° 12

Si realizzi una funzione di nome `scambiaInt`, che scambi il contenuto di due variabili intere, Si faccia un esempio di chiamata della funzione, per scambiare il contenuto delle variabili intere `x` e `y`.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

scambiaInt

Dati d'input: puntatori a due valori interi

Dati di output: valori scambiati

Variabili di lavoro: `temp`, per lo scambio

3. Costruzione del modello

4. Codifica

```
...  
void scambiaInt(int *p0, int *p1) {  
    int tmp;  
    tmp = *p0;  
    *p0 = *p1;  
    *p1 = tmp; }  
...  
scambiaInt(&x,&y);
```

5. Documentazione

6. Testing

PUNTATORI N° 13

Si scriva una funzione in grado di confrontare due stringhe, ritornando un risultato compatibile con la funzione strcmp, salvo il fatto che si devono ignorare le differenze tra caratteri maiuscoli e minuscoli. Si utilizzi una scansione delle stringhe basata su puntatori.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

confrontaStr

Dati d'input: puntatori a due stringhe

Dati di output: confronto stringhe: 0 se sono uguali, -1 se la prima è alfabeticamente minore della seconda, 1 viceversa.

Costanti:

Variabili di lavoro: *p0 e *p1 per gestire le stringhe ricevute in input.

3. Costruzione del modello

La soluzione proposta utilizza la funzione di libreria toupper (dato un carattere, se è alfabetico, ne ritorna la versione maiuscola). Si potrebbe utilizzare, in modo analogo, la tolower

4. Codifica

```
void confrontaStr(char *s0, char *s1) {  
    char *p0, *p1;  
    p0 = s0; p1 = s1;  
    while (toupper(*p0)!=toupper(*p1) && *p0!='\0' && *p1!='\0')  
    {  
        p0++; p1++;  
    }  
    return(toupper(p0)-toupper(p1));  
}
```

5. Documentazione

6. Testing

PUNTATORI N° 14

Dopo aver caricato in un vettore di stringhe n parole, scrivere un programma che, usando i sottoprogrammi, effettui le seguenti operazioni: cercare una parola scelta dall'utente, stampare l'elenco delle parole ordinato e non.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

carica

Dati d'input: *vocab*, vettore di stringhe

Dati di output:

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: *i*, indice ciclo

stampa

Dati d'input: *vocab*, vettore di stringhe

Dati di output: *vocab*, vettore di stringhe

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: *i*, indice ciclo

cerca

Dati d'input: *vocab*, vettore di stringhe

Dati di output: numero di parole cercate, parola cercata

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: *i*, *cerca*, *trovata*, *contatore* stringhe, *stringa* da cercare, *flag* per trovare la parola

ordina

Dati d'input: **vocab*, *puntaore* a vettori di stringhe *vettore* di stringhe

Dati di output:

Costanti: N numero massimo di stringhe

Variabili di lavoro: *i*, *j*, per spazzolare il vettore di stringhe, *appo*, vettore di char

main

Dati d'input:

Dati di output:

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: **p*, *puntatore* al vettore di stringhe, *i* assegnare ad ogni puntatore del vettore di puntatori una stringa del vettore di stringhe

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#define N 5
```

```
#define M 20
```

```
void carica (char vocab[N][M]);
```

```
void cerca(char vocab[N][M]);
```

```
void stampa(char (*vocab)[M]);
```

```
void ordina (char *vocab[]);
```

```
int main(void)
```

```
{ int i;
```

```
    char vocab[N][M];
```

```

char *p[N];
system("cls");
carica(vocab);
/* inizializzazione del vettore di puntatori a stringhe */
for (i=0;i<N;i++) p[i]=vocab[i];
stampa(vocab);
ordina (p);
printf("\n\ stampa vettore dopo ordinamento\n");
stampa(vocab);
cerca(vocab);
system("pause");
return(0);
}

void carica(char vocab[N][M])
{ /* Acquisisce parole da inserire nel vocabolario */
int i;//var locale
for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    gets(vocab[i]); }
}

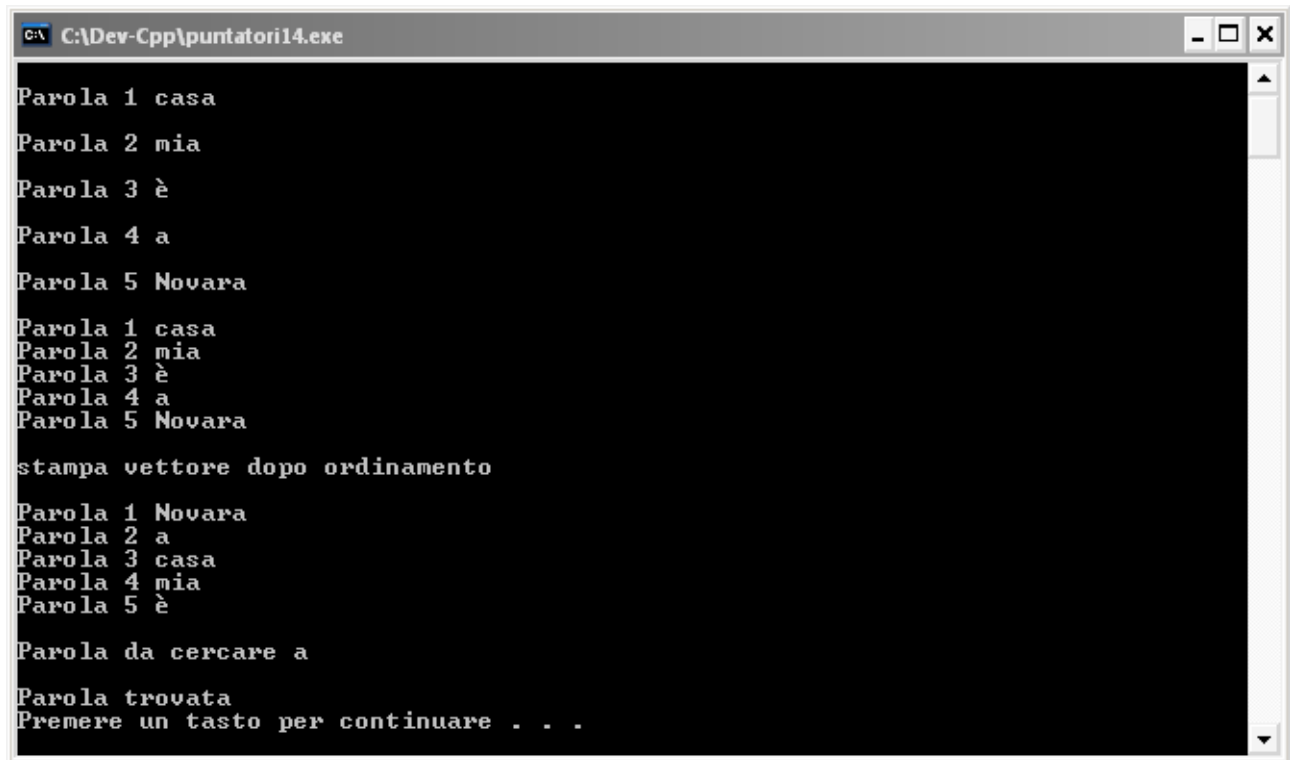
/* Acquisisce la parola da cercare e la cerca */
void cerca(char vocab[N][M])
{int i, trovata;
    char parcerc[M];
    i=0;
    printf("\n\ Parola da cercare ");
    gets(parcerc);
    trovata=0;
    while((i<N) &&(trovata==0)) {
        if (strcmp(parcerc,vocab[i])==0)
            trovata=1;
            i++; }
    if (trovata)
        printf("\nParola trovata\n");
    else
        printf("\nParola non trovata\n");
}

void stampa(char (*vocab)[M]) {
int i;
for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    printf("%s", vocab[i]);
}}

void ordina(char *vocab[N])
{
int i, j;
char appo[M];
for (i=0;i<N-1;i++)
    for (j=i+1;j<N;j++)
        if (strcmp(vocab[j],vocab[i])<0)
            {
                strcpy(appo, vocab[i]);
                strcpy(vocab[i], vocab[j]);
                strcpy(vocab[j], appo);
            }
}

```

```
}  
}
```



```
C:\Dev-Cpp\puntatori14.exe  
Parola 1 casa  
Parola 2 mia  
Parola 3 è  
Parola 4 a  
Parola 5 Novara  
Parola 1 casa  
Parola 2 mia  
Parola 3 è  
Parola 4 a  
Parola 5 Novara  
stampa vettore dopo ordinamento  
Parola 1 Novara  
Parola 2 a  
Parola 3 casa  
Parola 4 mia  
Parola 5 è  
Parola da cercare a  
Parola trovata  
Premere un tasto per continuare . . .
```

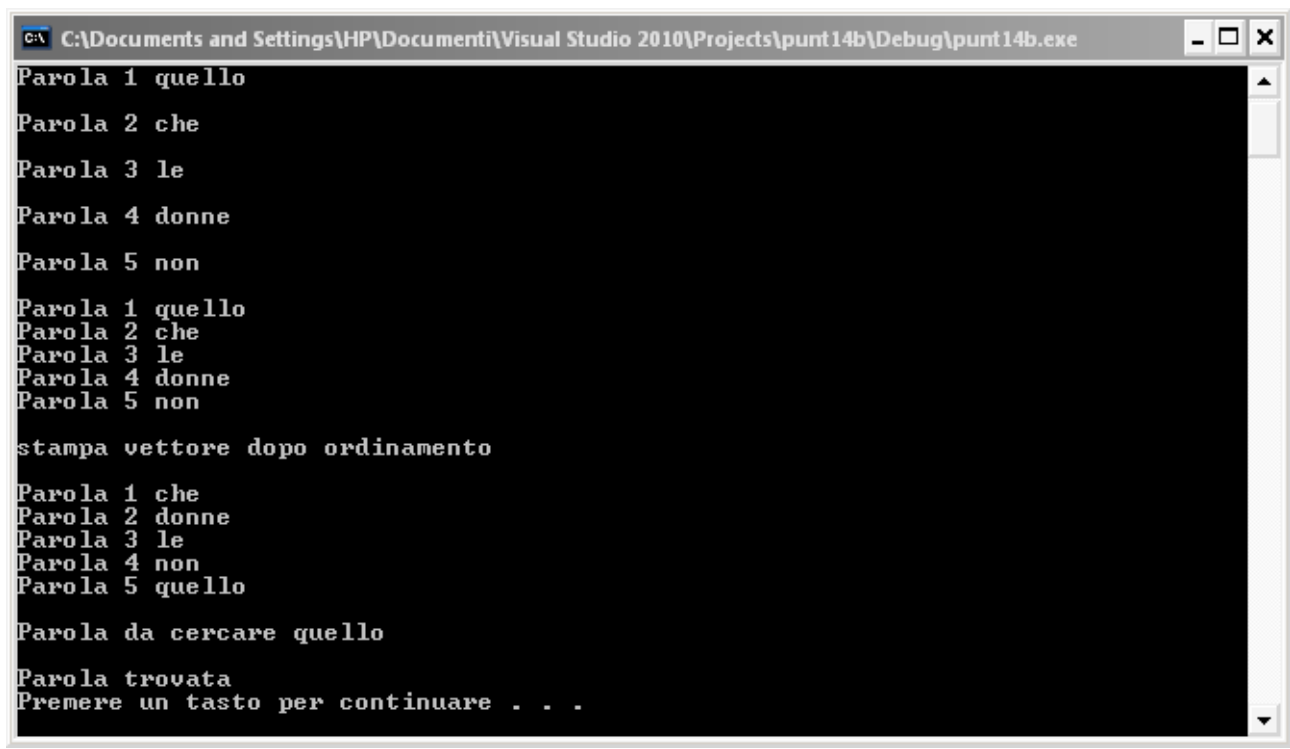
Microsoft Visual C++

```
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
#define N 5  
#define M 20  
void carica (char vocab[N][M]);  
void cerca(char vocab[N][M]);  
void stampa(char (*vocab)[M]);  
void ordina (char *vocab[]);  
int main(void)  
{ int i;  
  char vocab[N][M];  
  char *p[N];  
  system("cls");  
  carica(vocab);  
  /* inizializzazione del vettore di puntatori alle stringhe */  
  for (i=0;i<N;i++) p[i]=vocab[i];  
  stampa(vocab);  
  ordina (p);  
  printf("\n\nstampa vettore dopo ordinamento\n");  
  stampa(vocab);  
  cerca(vocab);  
  system("pause");  
  return(0);  
}  
void carica(char vocab[N][M])  
{ /* Acquisisce parole da inserire nel vocabolario */  
  int i;//var locale
```

```

for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    gets(vocab[i]);
}
}
/* Acquisisce la parola da cercare e la cerca */
void cerca(char vocab[N][M])
{int i, trovata;
    char parcerc[M];
    i=0;
    printf("\nParola da cercare ");
    gets(parcerc);
    trovata=0;
    while((i<N) &&(trovata==0)) {
        if (strcmp(parcerc,vocab[i])==0)
            trovata=1;
            i++; }
    if (trovata)
        printf("\nParola trovata\n");
    else
        printf("\nParola non trovata\n");
}
void stampa(char (*vocab)[M]) {
    int i;
    for (i=0;i<N;i++)
    {
        printf("\nParola %d ",i+1);
        printf("%s", vocab[i]);
    }
}
void ordina(char *vocab[N])
{
    int i, j;
    char appo[M];
    for (i=0;i<N-1;i++)
        for (j=i+1;j<N;j++)
            if(strcmp(vocab[j],vocab[i])<0)
            {
                strcpy(appo, vocab[i]);
                strcpy(vocab[i], vocab[j]);
                strcpy(vocab[j], appo);
            }
}
}

```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt14b\Debug\punt14b.exe
Parola 1 quello
Parola 2 che
Parola 3 le
Parola 4 donne
Parola 5 non
Parola 1 quello
Parola 2 che
Parola 3 le
Parola 4 donne
Parola 5 non
stampa vettore dopo ordinamento
Parola 1 che
Parola 2 donne
Parola 3 le
Parola 4 non
Parola 5 quello
Parola da cercare quello
Parola trovata
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

PUNTATORI N° 15

Dopo aver caricato in un vettore di stringhe n parole, scrivere un programma che, usando i sottoprogrammi, effettui le seguenti operazioni: contare il numero di volte che una parola scelta dall'utente, compare nel vettore, stampare l'elenco delle parole, cercare le parole che iniziano con una lettera scelta dall'utente e stamparle.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

carica

Dati d'input: vocab, vettore di stringhe

Dati di output:

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: i, indice ciclo

stampa

Dati d'input: vocab, vettore di stringhe

Dati di output:

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: i, indice ciclo

cercalett

Dati d'input: vocab, vettore di stringhe

Dati di output: numero di parole che iniziano con una lettera

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: i, c, lettcerca, contatore ciclo while, contatore parole che iniziano con lettera scelta, lettera iniziale scelta.

contaparola

Dati d'input: vocab, vettore di stringhe

Dati di output: numero stringhe che corrispondono alla stringa scelta

Costanti: N, M, numero massimo di stringhe, lunghezza stringa massima

Variabili di lavoro: i, c, cercapar, contatore ciclo while, contatore parole, parola da cercare

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#include <string.h>
```

```
#define N 10
```

```
#define M 20
```

```
void carica(char vocab[N][M]);
```

```
void cercalett(char vocab[N][M]);
```

```
void stampa(char vocab[N][M]);
```

```
void contaparola(char vocab[N][M]);
```

```
int main(void)
```

```
{ char vocab[N][M];
```

```
  system("cls");
```

```
  carica(vocab);
```

```
  stampa(vocab);
```

```
  contaparola(vocab);
```

```
  cercalett(vocab);
```

```
  system("pause");
```

```

return(0);}

void carica(char vocab[N][M])
{ /* Acquisisce parole da inserire nel vocabolario */
int i;//var locale
for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    gets(vocab[i]); } }
void contaparola(char vocab[N][M])
{ /*Acquisisce parola e conta il numero di volte che compare nell'elenco*/
int i,c;//var locali
    char cercapar[M];
    c=0; /*inizializzazione contatore parola trovata*/
    i=0; /*inizializzazione contatore ciclo*/
    printf("\nParola da cercare ");
    gets(cercapar);
while(i<N) {
    if (strcmp(cercapar,vocab[i])==0){ //confronta parola inserita con quelle dell'elenco
        c++; }
        i++; }
    if (c==0)
        printf("\nParola non trovata\n");
    else
        printf("\nParola trovata %d volte\n",c); }
/* Acquisisce la lettera da cercare e stampa le parole che iniziano con quella lettera*/
void cercalett(char vocab[N][M])
{int i,c;
    char lettcerca;
    c=0;
    i=0;
    printf("lettera da cercare ");
    lettcerca=getchar();
while(i<N) {
    if (lettcerca==vocab[i][0]){ /*confronta la lettera con quella iniziale delle parole inserite*/
        puts(vocab[i]);
        c++; }
        i++; }
    if (c==0)
        printf("\nParola non trovata\n");
    else
        printf("\nParole trovate che iniziano con la lettera %c sono: %d\n",lettcerca,c); }

void stampa(char vocab[N][M]) {
int i;
for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    puts(vocab[i]); } }

```

```
C:\Dev-Cpp\puntatori15.exe

Parola 1 e
Parola 2 dalle
Parola 3 macchine
Parola 4 per
Parola 5 noi
Parola 6 i complimenti
Parola 7 dei
Parola 8 playboy
Parola 9 noi
Parola 10 non
Parola 1 e
Parola 2 dalle
Parola 3 macchine
Parola 4 per
Parola 5 noi
Parola 6 i complimenti
Parola 7 dei
Parola 8 playboy
Parola 9 noi
Parola 10 non
Parola da cercare li
Parola non trovata
lettera da cercare n
noi
noi
non

Parole trovate che iniziano con la lettera n sono: 3
Premere un tasto per continuare . . .
```

Microsoft Visual C++

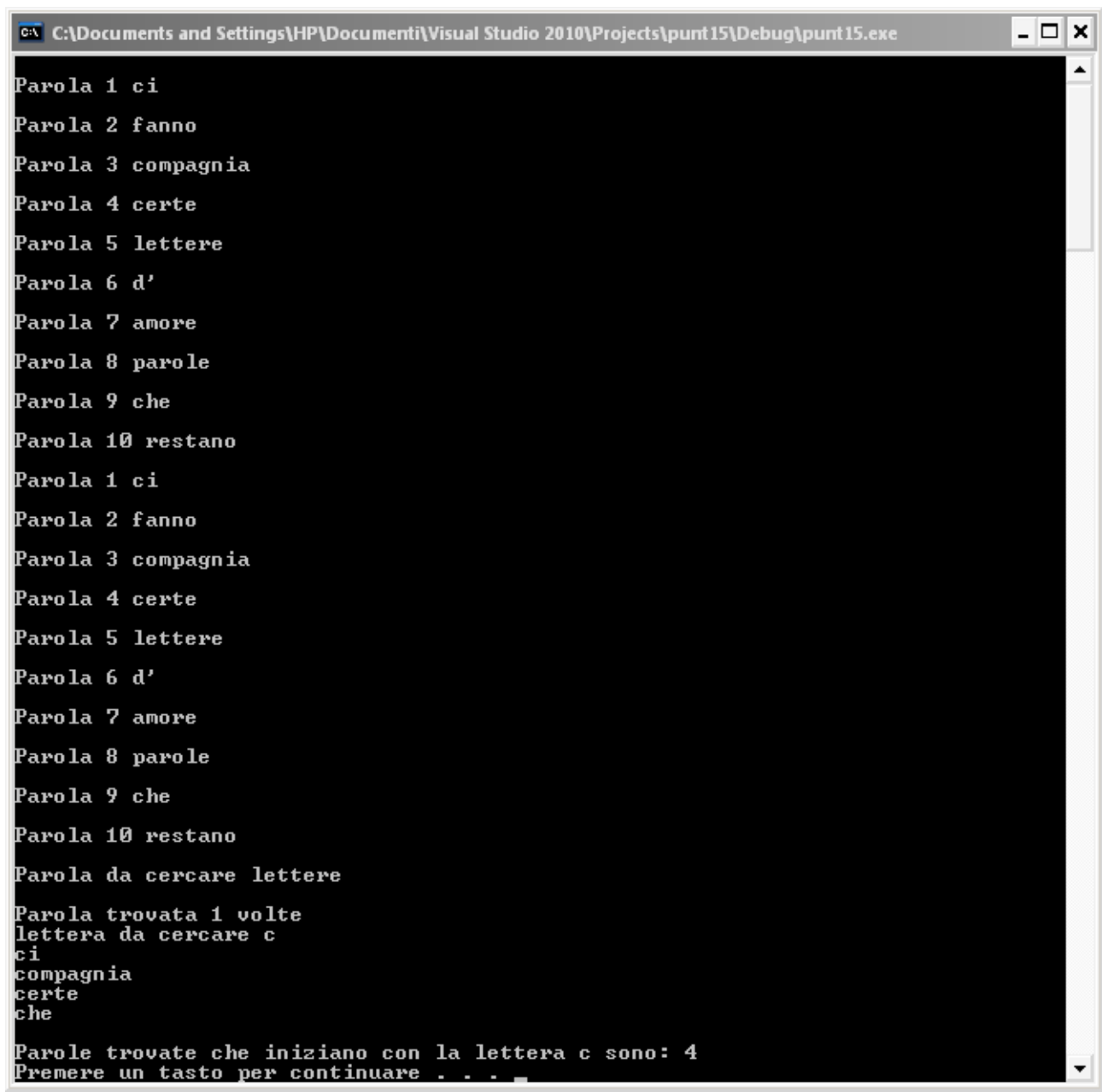
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 10
#define M 20
void carica (char vocab[N][M]);
void cercalett(char vocab[N][M]);
void stampa(char vocab[N][M]);
void contaparola(char vocab[N][M]);
int main(void)
{ char vocab[N][M];
  system("cls");
  carica(vocab);
  stampa(vocab);
  contaparola(vocab);
  cercalett(vocab);
  system("pause");
```



```

return(0);}
void carica(char vocab[N][M])
{ /* Acquisisce parole da inserire nel vocabolario */
int i;//var locale
for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    gets(vocab[i]); }
void contaparola(char vocab[N][M])
{ /* Acquisisce parola e conta il numero di volte che compare nell'elenco */
int i,c;//var locali
    char cercapar[M];
    c=0; /*inizializzazione contatore parola trovata*/
    i=0; /*inizializzazione contatore ciclo*/
    printf("\nParola da cercare ");
    gets(cercapar);
while(i<N) {
    if (strcmp(cercapar,vocab[i])==0){ /*confronta parola inserita con quelle dell'elenco
        c++; }
        i++; }
    if (c==0)
        printf("\nParola non trovata\t");
    else
        printf("\nParola trovata %d volte\n",c); }
/* Acquisisce la lettera da cercare e stampa le parole che iniziano con quella lettera */
void cercalett(char vocab[N][M])
{int i,c;
    char lettcerca;
    c=0;
    i=0;
    printf("\nlettera da cercare ");
    lettcerca=getchar();
while(i<N) {
    if (lettcerca==vocab[i][0]){ /*confronta la lettera con quella iniziale delle parole inserite
        puts(vocab[i]);
        c++; }
        i++;}
    if (c==0)
        printf("\nParola non trovata\t");
    else printf("\nParole trovate che iniziano con la lettera %c sono: %d\n",lettcerca,c); }
void stampa(char vocab[N][M]) {
int i;
for (i=0;i<N;i++) {
    printf("\nParola %d ",i+1);
    puts(vocab[i]); } }

```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt15\Debug\punt15.exe

Parola 1 ci
Parola 2 fanno
Parola 3 compagnia
Parola 4 certe
Parola 5 lettere
Parola 6 d'
Parola 7 amore
Parola 8 parole
Parola 9 che
Parola 10 restano
Parola 1 ci
Parola 2 fanno
Parola 3 compagnia
Parola 4 certe
Parola 5 lettere
Parola 6 d'
Parola 7 amore
Parola 8 parole
Parola 9 che
Parola 10 restano
Parola da cercare lettere
Parola trovata 1 volte
lettera da cercare c
ci
compagnia
certe
che
Parole trovate che iniziano con la lettera c sono: 4
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

PUNTATORI N° 16

Si sviluppi un programma che a partire dagli elementi contenuti in due array di interi ordinati costruisce un terzo array ordinato. Per esempio, se i due array di input contengono i seguenti valori: $a1 = \{3, 4, 5, 10, 12\}$, $a2 = \{1, 2, 6, 11, 18\}$, l'array risultante conterrà i seguenti valori: $ris = \{1, 2, 3, 4, 5, 6, 10, 11, 12, 18\}$. Si utilizzino i sottoprogrammi.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

merge

Dati d'input: a,b, c, primo vettore, secondo vettore, terzo vettore usato per la fusione.

Dati in output: i, vet, indice che spazzola il vettore, cella che contiene l'età.

Costanti: MDIM1, MDIM2,MDIM3, dimensione del primo vettore, dimensione del secondo vettore, dimensione del terzo vettore.

Variabili di lavoro: i, j, K, indici che spazzolano i vettori i,.

main

Dati di input

Dati in output:a, b, c, primo vettore, secondo vettore, vettore della fusione tra a e b

Costanti: MDIM1, MDIM2,MDIM3, dimensione del primo vettore, dimensione del secondo vettore, dimensione del terzo vettore.

Variabili di lavoro: i, a, b, c, primo vettore, secondo vettore, terzo vettore usato per la fusione, indice che spazzola il vettore

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include<stdlib.h>
```

```
#define MDIM1 10
```

```
#define MDIM2 10
```

```
#define MDIM3 15
```

```
void merge(int a[MDIM1],int b[MDIM2], int c[MDIM3]);
```

```
int main(void)
```

```
{ /* a e b sono gli array usati come input dalla funzione di merge. Si suppone che essi contengano valore ordinati in modo crescente. Si noti che gli array sono stati inizializzati al solo scopo di effettuare un test del sottoprogramma merge */
```

```
int a[MDIM1] = {2,5,6,7,11,12,18,20,21,26};
```

```
int b[MDIM2] = {0,1,3,8,9,15,17,19,22,23};
```

```
/* c e' l'array usato dalla funzione di merge per contenere il risultato dell'operazione */
```

```
int c[MDIM3];
```

```
int i;
```

```
merge(a,b,c);
```

```
printf("Primo array\n");
```

```
for(i=0;i<MDIM1;i++)
```

```
printf("%d, ", a[i]);
```

```
printf("\nSecondo array\n");
```

```
for(i=0;i<MDIM2;i++)
```

```
printf("%d, ", b[i]);
```

```
printf("\nArray risultante\n");
```

```
for(i=0;i<MDIM3;i++)
```

```
printf("%d, ", c[i]);
```

```

printf("\n");
system("pause");
return(0);}

void merge(int a[MDIM1],int b[MDIM2], int c[MDIM3]){
int i, j, k;
i = 0;
j = 0;
k = 0;
while (k<MDIM3 && i<MDIM1 && j<MDIM2){
if(a[i]<b[j]){
c[k] = a[i];
i++;}
else{
c[k] = b[j];
j++;}
k++;}
while(i<MDIM1 && k<MDIM3){
c[k] = a[i];
i++;
k++;}
while(j<MDIM2 && k<MDIM3){
c[k] = b[j];
j++;
k++;
}}

```

```

C:\Dev-Cpp\puntatori16.exe
Primo array
2, 5, 6, 7, 11, 12, 18, 20, 21, 26,
Secondo array
0, 1, 3, 8, 9, 15, 17, 19, 22, 23,
Array risultante
0, 1, 2, 3, 5, 6, 7, 8, 9, 11, 12, 15, 17, 18, 19,
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

#include <stdio.h>
#include<stdlib.h>
#define MDIM1 10
#define MDIM2 10
#define MDIM3 15
void merge(int a[MDIM1],int b[MDIM2], int c[MDIM3]);
int main(void)
{ /* a e b sono gli array usati come input dalla funzione di merge. Si suppone che essi contengano valore
ordinati in modo crescente. Si noti che gli array sono stati inizializzati al solo scopo di effettuare un test del
sottoprogramma merge */
int a[MDIM1] = {2,5,6,7,11,12,18,20,21,26};
int b[MDIM2] = {0,1,3,8,9,15,17,19,22,23};
/* c e' l'array usato dalla funzione di merge per contenere il risultato dell'operazione */
int c[MDIM3];
int i;
merge(a,b,c);
printf("Primo array\n");
for(i=0;i<MDIM1;i++)

```

```

printf("%d, ", a[i]);
printf("\nSecondo array\n");
for(i=0;i<MDIM2;i++)
printf("%d, ", b[i]);
printf("\nArray risultante\n");
for(i=0;i<MDIM3;i++)
printf("%d, ", c[i]);
printf("\n");
system("pause");
return(0);}

void merge(int a[MDIM1],int b[MDIM2], int c[MDIM3]){
int i, j, k;
i = 0;
j = 0;
k = 0;
while (k<MDIM3 && i<MDIM1 && j<MDIM2){
if(a[i]<b[j]){
c[k] = a[i];
i++;}
else{
c[k] = b[j];
j++;}
k++;}
while(i<MDIM1 && k<MDIM3){
c[k] = a[i];
i++;
k++;}
while(j<MDIM2 && k<MDIM3){
c[k] = b[j];
j++;
k++;}}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\punt16\Debug\punt16.exe
Primo array
2, 5, 6, 7, 11, 12, 18, 20, 21, 26,
Secondo array
0, 1, 3, 8, 9, 15, 17, 19, 22, 23,
Array risultante
0, 1, 2, 3, 5, 6, 7, 8, 9, 11, 12, 15, 17, 18, 19,
Premere un tasto per continuare . . . _

```

5. Documentazione

6. Testing

ALLOCAZIONE DINAMICA N° 1

Si scriva una funzione in grado di creare (e ritornare) una stringa data dalla concatenazione di due stringhe, ricevute come parametri. La stringa generata va allocata dinamicamente.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Dati d'input: *s0, *s1: puntatori a due stringhe

Dati in output: s01: stringa concatenata

Variabili di lavoro: *s01 puntatore alla stringa che contiene il contenuto della concatenazione

3. Costruzione del modello

La soluzione proposta utilizza le funzioni di libreria strcpy e strcat, in grado, rispettivamente, di copiare una stringa e di appendere una stringa in fondo ad un'altra.

4. Codifica

```
.....  
char *concatenaStr(char *s0, char *s1) {  
    char *s01;  
    s01 = malloc((strlen(s0)+strlen(s1)+1)*sizeof(char));  
    strcpy(s01,s0);  
    strcat(s01,s1);  
    return(s01); }
```

5. Documentazione

6. Testing

ALLOCAZIONE DINAMICA N° 2

Si scriva una funzione `malloc2d`, in grado di allocare una matrice rettangolare di numeri reali (tipo `float`), le cui dimensioni sono ricevute come parametri. La matrice viene inizializzata azzerando tutte le caselle.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Dati d'input: `nr`, `nc`: numero righe, numero colonne

Dati in output: `m`: matrice

Variabili di lavoro: `**m` puntatore alla matrice

3. Costruzione del modello

La soluzione proposta alloca un vettore di puntatori a righe, ognuna delle quali viene successivamente allocata e inizializzata.

4. Codifica

```
.....  
float **malloc2d(int nr, int nc) {  
    float **m;  
    int i,j;  
    m = malloc(nr*sizeof(float *));  
    for (i=0; i<nr; i++) {  
        m[i] = malloc(nc*sizeof(float));  
        for (j=0; j<nc; j++) {  
            m[i][j] = 0.0; } }  
    return(m); }
```

5. Documentazione

6. Testing

ALLOCAZIONE DINAMICA N° 3

Sia dato un vettore di interi presente in memoria (allocato dinamicamente). Si scriva una funzione C che ricevendo il vettore e la sua dimensione restituisca la media degli interi presenti nel vettore non considerando gli eventuali duplicati. Ad esempio, se il vettore contiene gli interi 7 6 4 6 la funzione deve restituire il valore 5.6, ovvero la media di 7, 6 e 4.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

gia_presente

Dati d'input: v, e: vettore e indice dell' elemento già presente

Dati in output: 1, se elemento già presente, altrimenti 0

Variabili di lavoro: j, indice del ciclo

Media

Dati d'input: vet, n: vettore, numero elementi

Dati in output: somma/conta, media elementi senza duplicati

Variabili di lavoro: somma, conta, i, somma elementi nuovi, conta elementi non doppi, indice del ciclo

main

Dati d'input:

Dati in output: 1, se elemento già presente, altrimenti 0

Variabili di lavoro: *p, i, n, puntatore al vettore, indice per spazzolare il vettore, numero elementi

3. Costruzione del modello

La funzione Media() calcola la somma (e conta) i valori presenti nel vettore che non siano già presenti nella parte precedentemente analizzata del vettore stesso. Quest'ultimo controllo è affidato alla funzione gia_presente().

4. Codifica

Dev-C++ 4.9.9.2


```
#include <stdio.h>
#include <stdlib.h>
int gia_presente(int v[], int e);
float Media(int vet[], int n);
int main(void){
    int i,n;
    int *p;
    printf("Numero di elementi da inserire:\n ");
    scanf("%d",&n);
    p = (int*) malloc(n * sizeof(int));
    for (i = 0; i < n; i++)
        scanf("%d",&p[i]);
    printf("La media senza duplicati e' %f\n",Media(p,n));
    system ("pause");
    return(0);}
int gia_presente(int v[], int e){/* funzione ausiliaria: controlla se l'elemento di indice e e'
presente nella parte del vettore v tra 0 ed e-1 */
    int j;
    for (j = 0; j < e; j++)
        if (v[j] == v[e])
            return 1;
```



```

return 0;}
float Media(int vet[], int n){
int somma = 0, conta = 0, i;
for (i = 0; i < n; i++)
if (!gia_presente(vet,i)){
somma += vet[i];
conta++;}
return ((float) somma) / conta;}

```



```

C:\Dev-Cpp\allocaldinam3.exe
Numero di elementi da inserire:
5
1
2
3
5
7
La media senza duplicati e' 3.600000
Premere un tasto per continuare . . .

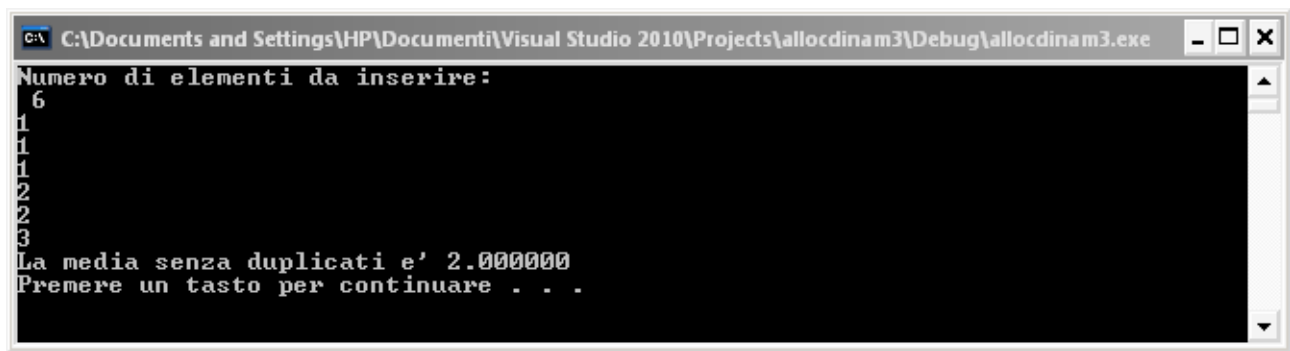
```

Microsoft Visual C++

```

#include <stdio.h>
#include<stdlib.h>
int gia_presente(int v[], int e);
float Media(int vet[], int n);
int main(void){
int i,n;
int *p;
printf("Numero di elementi da inserire:\n ");
scanf("%d",&n);
p = (int*) malloc(n * sizeof(int));
for (i = 0; i < n; i++)
scanf("%d",&p[i]);
printf("La media senza duplicati e' %f\n",Media(p,n));
system ("pause");
return(0);}
int gia_presente(int v[], int e){ /* funzione ausiliaria: controlla se l'elemento di indice e e' presente nella
parte del vettore v tra 0 ed e-1 */
int j;
for (j = 0; j < e; j++)
if (v[j] == v[e])
return 1;
return 0;}
float Media(int vet[], int n){
int somma = 0, conta = 0, i;
for (i = 0; i < n; i++)
if (!gia_presente(vet,i)){
somma += vet[i];
conta++;}
return ((float) somma) / conta;}

```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\allocdinam3\Debug\allocdinam3.exe
Numero di elementi da inserire:
6
1
1
1
1
2
2
3
La media senza duplicati e' 2.000000
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

ALLOCAZIONE DINAMICA N° 4

Caricare e stampare un vettore la cui dimensione è scelta in modo dinamico.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

carica_stamp

Dati d'input: *p, n: puntatore al vettore e numero di elementi da allocare dinamicamente

Dati in output: p, vettore allocato in modo dinamico

Variabili di lavoro: i, indice per spazzolare gli elementi del vettore

main

Dati d'input:

Dati in output:

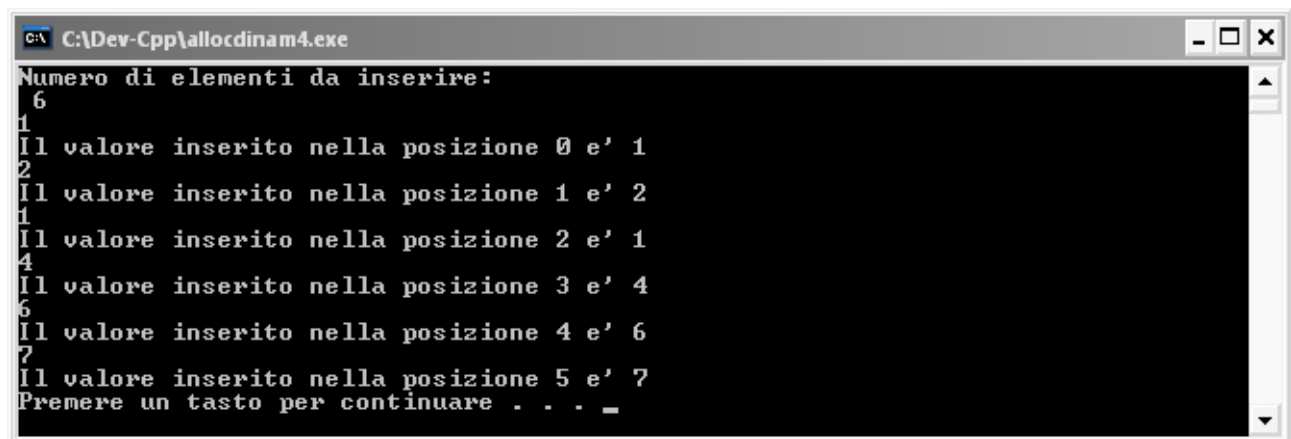
Variabili di lavoro: *p, n, puntatore al vettore, numero elementi

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
void carica_stamp(int *p, int n);
int main(void){
    int n;
    int *p;
    printf("Numero di elementi da inserire:\n ");
    scanf("%d",&n);
    carica_stamp(p,n);
    free (p);          /*libera memoria allocata*/
    system ("pause");
    return(0);}
void carica_stamp (int *p, int n){int i;
p = (int*) malloc(n * sizeof(int));
for (i = 0; i < n; i++){
    scanf("%d",&p[i]);
    printf("Il valore inserito nella posizione %d e' %d\n", i, p[i]);
}
}
```



```
C:\Dev-Cpp\allocdinam4.exe
Numero di elementi da inserire:
6
1
Il valore inserito nella posizione 0 e' 1
2
Il valore inserito nella posizione 1 e' 2
1
Il valore inserito nella posizione 2 e' 1
4
Il valore inserito nella posizione 3 e' 4
6
Il valore inserito nella posizione 4 e' 6
7
Il valore inserito nella posizione 5 e' 7
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

```

#include <stdio.h>
#include <stdlib.h>
void carica_stampa(int *p, int n);
int main(void){
    int n;
    int *p;
    printf("Numero di elementi da inserire:\n ");
    scanf("%d",&n);
    p = (int*) malloc(n * sizeof(int));
    carica_stampa(p,n);
    free (p);          /*libera memoria allocata*/
    system ("pause");
    return(0);}
void carica_stampa (int *p, int n){int i;

    for (i = 0; i < n; i++){
        scanf("%d",&p[i]);
        printf("Il valore inserito nella posizione %d e' %d\n",i, p[i]);
    }
}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\allocdinam4\Debug\allocdinam4.exe
Numero di elementi da inserire:
5
2
Il valore inserito nella posizione 0 e' 2
3
Il valore inserito nella posizione 1 e' 3
4
Il valore inserito nella posizione 2 e' 4
5
Il valore inserito nella posizione 3 e' 5
6
Il valore inserito nella posizione 4 e' 6
Premere un tasto per continuare . . . _

```

5. Documentazione

6. Testing

RECORD N° 1

Si implementi un **sottoprogramma** che riceve i seguenti parametri: una struct contenente un array di interi a e un valore intero b di cui si vogliono determinare le occorrenze nel vettore.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

conta

Dati d'input: *arch, b, record di tipo **archivio** e numero da contare

Dati in output: cont, numero di occorrenze di b

Costanti: N, dimensione del vettore

Variabili di lavoro: cont, i, numero di occorrenze di b, indice che spazzola il vettore contenuto nel record

carica

Dati d'input: *arch, puntatore al record

Dati in output:

Costanti: N, dimensione del vettore.

Variabili di lavoro: i, indice che spazzola il vettore contenuto nel record

stampa

Dati d'input: *arch, puntatore al record

Dati in output: arch->a[i], elementi del vettore contenuto nel record di tipo **archivio**

Costanti: N, dimensione del vettore.

Variabili di lavoro: i, indice che spazzola il vettore contenuto nel record

main

Dati d'input:

Dati in output: occorrenze di b

Costanti: N, dimensione dei vettori.

Variabili di lavoro: i, indice che spazzola il vettore contenuto nel record, *p, record di tipo **archivio** (struct contenente vettore **a** di numeri e **num**, numero di elementi e numero da contare), b numero da contare, arch record di tipo **archivio**

3. Costruzione del modello

Il sottoprogramma calcola il numero di occorrenze del valore intero b nell'array a e ritorna al chiamante il valore trovato. Per esempio, se l'array contiene i seguenti valori: {0, 2, 3, 0, 3, 5} ed il valore intero b passato al sottoprogramma è 3, il risultato che il sottoprogramma dovrà produrre e passare al chiamante è 2.

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define N 6
```

```
struct archivio{
```

```
int a[N];
```

```
int num;};
```

```
void carica(archivio *arch);
```

```
void stampa(archivio *arch);
```

```
int conta(archivio *arch, int b);
```

```
int main (void)
```

```
{int b;
```

```
archivio arch;
```

```
archivio *p;
```

```

p=&arch;
carica(p);           /*caricamento record*/
stampa(p);
printf("inserisci il numero di cui vuoi calcolare le occorrenze\n");
scanf("%d", &b);
printf("le occorrenze di %d sono %d\n",b, conta(p,b));
system("pause"); return(0);}
void carica(archivio *arch)
{arch->a[0]=0;
arch->a[1]=2;
arch->a[2]=3;
arch->a[3]=0;
arch->a[4]=2;
arch->a[5]=5;
arch->num=6;}
void stampa(archivio *arch){
int i;
for (i=0;i<arch->num;i++)
printf("%d\n", arch->a[i]);}
int conta(archivio *arch, int b)
{int i, cont;
cont = 0;
for(i=0;i<arch->num;i++)
if(arch->a[i] == b)
cont++;
return (cont);
}

```

```

C:\Dev-Cpp\record1.exe
0
2
3
0
2
5
inserisci il numero di cui vuoi calcolare le occorrenze
2
le occorrenze di 2 sono 2
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

#include <stdio.h>
#include <stdlib.h>
#define N 6
struct archivio{
int a[N];
int num;};
void carica(archivio *arch);
void stampa(archivio *arch);
int conta(archivio *arch, int b);
int main (void)
{int b;
archivio arch;
archivio *p;
p=&arch;

```

```

carica(p);          /*caricamento record*/
stampa(p);
printf("inserisci il numero di cui vuoi calcolare le occorrenze\n");
scanf("%d", &b);
printf("le occorrenze di %d sono %d\n",b, conta(p,b));
system("pause"); return(0);}
void carica(archivio *arch)
{arch->a[0]=0;
arch->a[1]=2;
arch->a[2]=3;
arch->a[3]=0;
arch->a[4]=2;
arch->a[5]=5;
arch->num=6;}
void stampa(archivio *arch){
int i;
for (i=0;i<arch->num;i++)
printf("%d\n", arch->a[i]);}
int conta(archivio *arch, int b)
{int i, cont;
cont = 0;
for(i=0;i<arch->num;i++)
if(arch->a[i] == b)
cont++;
return (cont);}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\record1\Debug\record1.exe
0
2
3
0
2
5
inserisci il numero di cui vuoi calcolare le occorrenze
5
le occorrenze di 5 sono 1
Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

RECORD N° 2

Si implementi un programma che riceve come parametro le informazioni relative ad un rettangolo, ne calcola l'area e restituisce tale valore al chiamante utilizzando il meccanismo del valore di ritorno.

1. Identificazione del sistema

Il sistema è di tipo matematico.

2. Analisi dei dati

carica

Dati d'input: *r, record contenente i lati del rettangolo

Dati in output:

Variabili di lavoro:

stampa

Dati d'input: *r, record contenente i lati rettangolo

Dati in output: r->l1, r->l2, base e altezza del rettangolo

Variabili di lavoro:

areaRett

Dati d'input: *r, record contenente i lati rettangolo

Dati in output: area del rettangolo

Variabili di lavoro

main

Dati d'input:

Dati in output: area del rettangolo

Variabili di lavoro *p, record contenente i lati rettangolo, memorizzati in **Rettangolo**

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
struct Rettangolo{
float lato1;
float lato2;};
void carica(struct Rettangolo *r);
void stampa(Rettangolo *r);
float areaRett(struct Rettangolo *r);
int main (void)
{struct Rettangolo r;
struct Rettangolo *p;
p=&r;
carica(p);
stampa(p); /*caricamento record*/
printf("l'area del rettangolo vale %f\n", areaRett(p));
system("pause"); return(0);}
```

```
void carica(struct Rettangolo *r)
{printf ("inserisci la base\n");
scanf ("%f",&r->lato1);
printf ("inserisci l'altezza\n");
scanf ("%f",&r->lato2);}
```



```
void stampa (struct Rettangolo *r)
{printf ("i lati del rettangolo sono %f %f\n", r->lato1, r->lato2);}
```

```
float areaRett(struct Rettangolo *r)
{return (r->lato1 * r->lato2);}
```

```
C:\Dev-Cpp\record2.exe
inserisci la base
34
inserisci l'altezza
87
i lati del rettangolo sono 34.000000 87.000000
l'area del rettangolo vale 2958.000000
Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
struct Rettangolo{
float lato1;
float lato2;};
void carica(struct Rettangolo *r);
void stampa(struct Rettangolo *r);
float areaRett(struct Rettangolo *r);
int main (void)
{struct Rettangolo r;
struct Rettangolo *p;
p=&r;
carica(p);
stampa(p); /*caricamento record*/
printf("l'area del rettangolo vale %f\n", areaRett(p));
system("pause"); return(0);}
void carica(struct Rettangolo *r)
{printf ("inserisci la base\n");
scanf ("%f",&r->lato1);
printf ("inserisci l'altezza\n");
scanf ("%f",&r->lato2);}
void stampa (struct Rettangolo *r)
{printf ("i lati del rettangolo sono %f %f\n", r->lato1, r->lato2);}
float areaRett(struct Rettangolo *r)
{return (r->lato1 * r->lato2);}
```

```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\record2\Debug\record2.exe
inserisci la base
20
inserisci l'altezza
60
i lati del rettangolo sono 20.000000 60.000000
l'area del rettangolo vale 1200.000000
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

RECORD N° 3

Data la seguente definizione di tipo:

```
typedef struct {  
    int N;  
    int num[10];  
} numeri;
```

Cosa fa la seguente funzione?

```
int XXX(numeri a)  
{  
    int i;  
    int cont;  
    cont = 0;  
    for(i = 0; i < a.N; i++)  
    {  
        if (a.num[i]%2 != 0)  
            cont++;  
    }  
    return(cont);  
}
```

Data la seguente situazione iniziale:

$a.N = 8$;

$a.num = \{12, 4, 0, 3, 5, 0, 3, 2, 0, 2\}$

Quale valore restituisce la funzione?

Soluzione

La funzione conta il numero di valori dispari (non divisibili per due) che sono presenti nell'array incapsulato nella struct numeri, e restituisce tale numero al chiamante. Nell'esempio riportato sopra, il valore restituito dalla funzione è 3.

RECORD N° 4

Si implementi un programma che tramite un'apposita funzione, raddoppi le coordinate delle variabili di tipo struct punto passate come parametro.

1. Identificazione del sistema

Il sistema è di tipo matematico.

2. Analisi dei dati

raddoppia

Dati d'input: *p, puntatore al record di tipo punto (struct contenente le coordinate di un punto nello spazio(x,y,z))

Dati in output: raddoppia le coordinate del punto

Variabili di lavoro:

main

Dati d'input:

Dati in output: area del rettangolo

Variabili di lavoro: p, record di tipo punto (struct contenente le coordinate di un punto nello spazio(x,y,z))

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
struct punto{
float x, y, z;};
void raddoppia(struct punto *p );
void stampa(struct punto p);
int main (void)
{struct punto p;
p.x = 3;
p.y = 4;
p.z =5;
stampa(p); /*caricamento record*/
raddoppia(&p);
stampa(p);
system("pause"); return(0);}
void raddoppia(struct punto *p)
{p->x *=2;
p->y *=2;
p->z *=2;}
void stampa (struct punto p)
{printf ("(%.2f , %.2f, %.2f)\n", p.x,p.y,p.z);
}
```

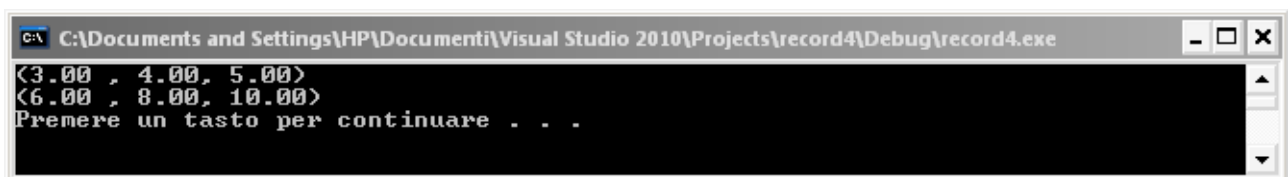


Microsoft Visual C++

```

#include <stdio.h>
#include <stdlib.h>
struct punto{
float x, y, z;};
void raddoppia(struct punto *p );
void stampa(struct punto p);
int main (void)
{struct punto p;
p.x = 3;
p.y = 4;
p.z =5;
stampa(p); /*caricamento record*/
raddoppia(&p);
stampa(p);
system("pause"); return(0);}
void raddoppia(struct punto *p)
{p->x *=2;
p->y *=2;
p->z *=2;}
void stampa (struct punto p)
{printf ("(%.2f , %.2f, %.2f)\n", p.x,p.y,p.z);
}

```



```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\record4\Debug\record4.exe
(3.00 , 4.00, 5.00)
(6.00 , 8.00, 10.00)
Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

LISTE LINEARI N° 1

Si vuole realizzare un tipo struct, utilizzato per informazioni su operazioni di vendita, avente i seguenti campi:

- codice: numero intero indicante il codice di riferimento dell'articolo venduto
- nome: stringa di lunghezza inferiore a 20 caratteri
- prezzo: numero reale (float) corrispondente al prezzo unitario dell'articolo
- npezzi: numero (intero) di pezzi venduti

Con tale tipo si vogliono realizzare liste dinamiche. La struct deve quindi essere realizzata come struttura ricorsiva, con puntatore a un dato dello stesso tipo. Si definiscano in C il tipo struct, utilizzando due diversi schemi: (a) una struct a un solo livello, contenente tutti i campi, puntatore ricorsivo compreso; (b) una struct a due livelli, nella quale a primo livello si accede a un puntatore ricorsivo e ad una sotto-struttura contenente il resto dei dati.

2. Analisi dei dati.

Schema (1): tipo operazione_t, dichiarato mediante typedef, contenente tutti i campi.

Schema (2): tipo operazione_p (puntatore) dichiarato prima di definire la struct (eccezione ammessa in C). Tipo operazione_t, dichiarato mediante typedef, contenente tutti i campi

Schema (3): tipo info_t per i dati, il resto secondo la strategia (2).

3. Costruzione del modello

4. Codifica

```
1. typedef struct operazione {  
    int codice;  
    char nome[20];  
    float prezzo;  
    int npezzi;  
    struct operazione *link; } operazione_t;  
2. typedef struct operazione *operazione_p;  
   typedef struct operazione {  
       int codice;  
       char nome[20];  
       float prezzo;  
       int npezzi;  
       operazione_p link; } operazione_t;  
3. typedef struct {  
    int codice;  
    char nome[20];  
    float prezzo;  
    int npezzi; } info_t;  
   typedef struct operazione *operazione_p; {  
       info_t dati;  
       operazione_p link; } operazione_t;
```

5. Documentazione

6. Testing

LISTE LINEARI N° 2

Si ridefinisca la struct dell'esercizio 1, utilizzando per il campo nome una stringa dinamica. Si scriva poi una funzione in grado di copiare il contenuto di una struttura in un'altra, duplicando la stringa nome. La struttura destinazione viene ricevuta "per indirizzo", mentre la sorgente viene ricevuta "per valore".

2. Analisi dei dati.

Schema (1): tipo operazione_t, dichiarato mediante typedef, contenente tutti i campi.

3. Costruzione del modello

Si utilizza la funzione strdup.

4. Codifica

```
typedef struct operazione {  
    int codice;  
    char *nome;  
    float prezzo;  
    int npezzi;  
    struct operazione *link;  
} operazione_t;  
  
...  
void copiaOperazione(operazione_t *dst, operazione_t src)  
{  
    *dst = src;  
    dst->nome = strdup(src.nome);  
}
```

5. Documentazione

6. Testing

LISTE LINERARI N° 3

Si definisca la struttura dati nel caso in cui la pila contenga degli interi e si implementino i sottoprogrammi push e pop.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

push:

Dati d'input: *p, elem, puntatore alla **pila** (struct contenente array di numeri e cursore) e elemento da inserire

Dati in output:

Costanti: DIM, dimensione massima pila

Variabili di lavoro: i, indice che spazzola il vettore contenuto nel record

pop:

Dati d'input: *p, puntatore alla pila da cui si vuole cancellare

Dati in output: elemento cancellato o -1 se pila vuota

Costanti: DIM, dimensione massima pila

main:

Dati d'input:

Dati in output:

Costanti: DIM, dimensione massima pila

Variabili di lavoro: scelta, numero, *P, per scegliere dal menù principale l'operazione da effettuare, valore da inserire o cancellare nella pila, puntatore alla pila

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define DIM 10 /* dimensione massima della pila */
```

```
/* definizione della struttura dati della pila. In questo caso l'array che serve per memorizzare gli elementi della pila ed il cursore sono incapsulati in una struct. Questa soluzione rende più evidente il collegamento logico esistente tra queste due parti */
```

```
typedef struct {
```

```
int array[DIM];
```

```
int cursore;
```

```
} pila;
```

```
void push(int elem, pila *p);
```

```
int pop(pila *p);
```

```
int main(void)
```

```
{pila P; /* dichiarazione della pila */
```

```
int scelta, numero; /* variabili utilizzate per la gestione dei menu */
```

```
P.cursore = 0;
```

```
scelta=0;
```

```
while(scelta!=3){
```

```
printf("*****\n");
```

```
printf("***** MENU PRINCIPALE *****\n");
```

```
printf("*****\n\n");
```

```
printf("\t1) Push\n");
```

```
printf("\t2) Pop\n");
```

```
printf("\t3) Esci\n\n");
```

```

printf("Cosa vuoi fare? ");
scanf("%d",&scelta);
switch(scelta){
case 1:printf("digita l'intero da inserire nella pila\n");
    scanf("%d", &numero);
    if (numero > 0) push(numero, &P);
    break;
case 2:numero = pop(&P);
    if (numero>0)
        printf("il numero prelevato dalla pila e' :%d\n",numero);
    else printf("la pila e' vuota\n");
    break;
case 3:printf("Programma terminato\n");
    break;
default:printf("Devi inserire un numero compreso tra 1 e 3!");
    break;}
printf("\n\n");}}
/* Riceve un intero come parametro e lo inserisce nella pila. Il parametro p e` il riferimento alla pila in cui
vengono inseriti gli elementi*/
void push(int elem, pila *p){
/* si noti l'uso del simbolo -> per accedere agli elementi della struct puntata dal puntatore p. Questa e` una
scorciatoia che ci permette di evitare l'uso della sintassi: (*p).cursore che risulta meno leggibile. E` sempre
possibile utilizzare il simbolo -> per accedere agli elementi di una struct puntata da un puntatore */
if (p->cursore < DIM){ /* e` possibile inserire un nuovo elemento nella pila */
p->array[p->cursore] = elem;
p->cursore = p->cursore+1; /* il cursore indica sempre la prima posizione libera in testa alla pila */
}
else printf("La pila e' piena\n");}
/* Restituisce un intero che e` l'elemento in testa alla pila .Il parametro p e` un riferimento alla pila */
int pop(pila *p){
if (p->cursore > 0){
p->cursore = p->cursore-1; /* il cursore viene decrementato in modo tale da liberare la posizione in
testa alla pila */
return (p->array[p->cursore]);}
else return (-1); /* la pila e` vuota. NB: si assume che i valori inseriti nella pila siano sempre positivi */
}

```



```
C:\Dev-Cpp\listalineare3.exe
*****
***** MENU PRINCIPALE *****
*****

1> Push
2> Pop
3> Esci

Cosa vuoi fare? 2
la pila e' vuota

*****
***** MENU PRINCIPALE *****
*****

1> Push
2> Pop
3> Esci

Cosa vuoi fare? 1
digita l'intero da inserire nella pila
23

*****
***** MENU PRINCIPALE *****
*****

1> Push
2> Pop
3> Esci

Cosa vuoi fare? 1
digita l'intero da inserire nella pila
22

*****
***** MENU PRINCIPALE *****
*****

1> Push
2> Pop
3> Esci

Cosa vuoi fare? 2
il numero prelevato dalla pila e' :22

*****
***** MENU PRINCIPALE *****
*****

1> Push
2> Pop
3> Esci

Cosa vuoi fare?
```

Microsoft Visual C++

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define DIM 10 /* dimensione massima della pila */
```

```
/* definizione della struttura dati della pila. In questo caso l'array che serve per memorizzare gli elementi della pila ed il cursore sono incapsulati in una struct. Questa soluzione rende più evidente il collegamento logico esistente tra queste due parti */
```

```
typedef struct {
```

```
int array[DIM];
```

```
int cursore;
```

```
} pila;
```

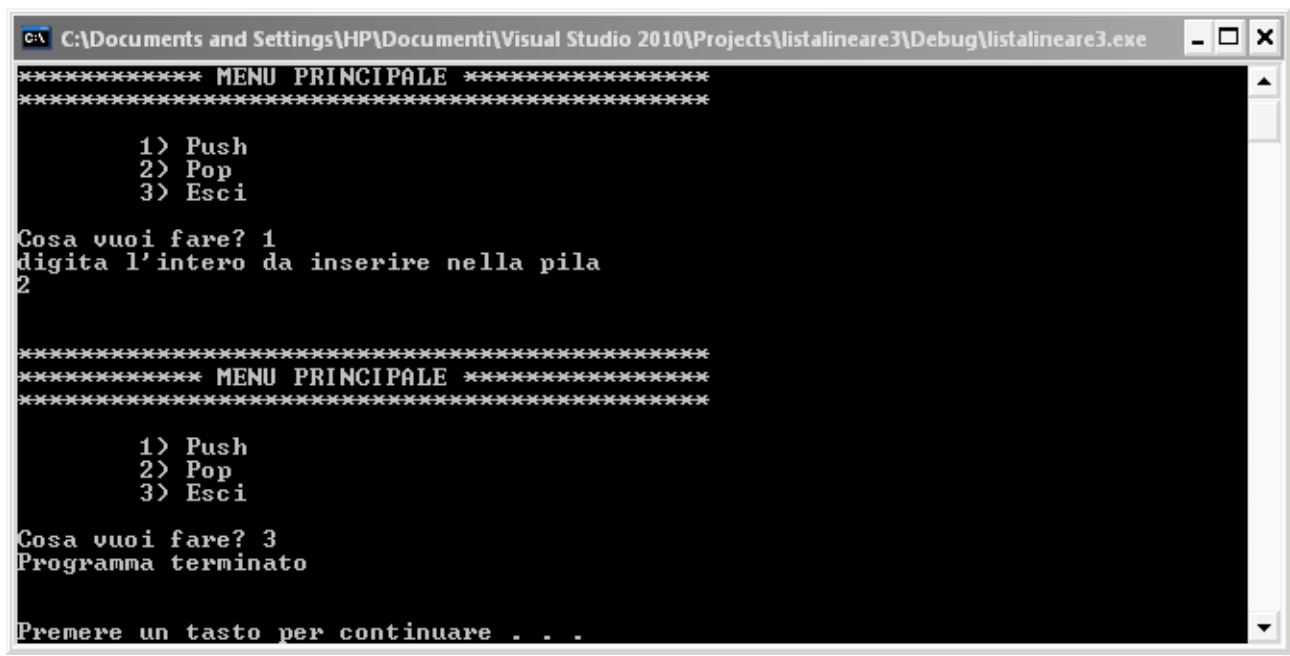
```
void push(int elem, pila *p);
```

```
int pop(pila *p);
```

```

int main(void)
{pila P; /* dichiarazione della pila */
int scelta, numero; /* variabili utilizzate per la gestione dei menu */
P.cursore = 0;
scelta=0;
while(scelta!=3){
printf("*****\n");
printf("***** MENU PRINCIPALE *****\n");
printf("*****\n\n");
printf("\t1) Push\n");
printf("\t2) Pop\n");
printf("\t3) Esci\n\n");
printf("Cosa vuoi fare? ");
scanf("%d",&scelta);
switch(scelta){
case 1:printf("digita l'intero da inserire nella pila\n");
scanf("%d", &numero);
if (numero > 0) push(numero, &P);
break;
case 2:numero = pop(&P);
if (numero>0)
printf("il numero prelevato dalla pila e' :%d\n",numero);
else printf("la pila e' vuota\n");
break;
case 3:printf("Programma terminato\n");
break;
default:printf("Devi inserire un numero compreso tra 1 e 3!");
break;}
printf("\n\n");system("pause"); return(0);}
/* Riceve un intero come parametro e lo inserisce nella pila. Il parametro p e` il riferimento alla pila in cui
vengono inseriti gli elementi*/
void push(int elem, pila *p){
/* si noti l'uso del simbolo -> per accedere agli elementi della struct puntata dal puntatore p. Questa e` una
scorciatoia che ci permette di evitare l'uso della sintassi: (*p).cursore che risulta meno leggibile. E` sempre
possibile utilizzare il simbolo -> per accedere agli elementi di una struct puntata da un puntatore */
if (p->cursore < DIM){ /* e` possibile inserire un nuovo elemento nella pila */
p->array[p->cursore] = elem;
p->cursore = p->cursore+1; /* il cursore indica sempre la prima posizione libera in testa alla pila */
}
else printf("La pila e' piena\n");}
/* Restituisce un intero che e` l'elemento in testa alla pila .Il parametro p e` un riferimento alla pila */
int pop(pila *p){
if (p->cursore > 0){
p->cursore = p->cursore-1; /* il cursore viene decrementato in modo tale da liberare la posizione in
testa alla pila */
return (p->array[p->cursore]);}
else return (-1); /* la pila e` vuota. NB: si assume che i valori inseriti nella pila siano sempre positivi */
}

```



A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\listalineare3\Debug\listalineare3.exe. The window contains a text-based menu program. It starts with a separator line of asterisks and the text 'MENU PRINCIPALE'. Below this, it lists three options: '1> Push', '2> Pop', and '3> Esci'. The program then prompts 'Cosa vuoi fare? 1' and 'digita l'intero da inserire nella pila'. The user has entered '2'. The program then displays the same menu again. This time, the user has entered '3'. The program responds with 'Programma terminato' and 'Premere un tasto per continuare . . .'. The window has standard Windows controls (minimize, maximize, close) in the top right corner.

```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\listalineare3\Debug\listalineare3.exe
***** MENU PRINCIPALE *****
*****
1> Push
2> Pop
3> Esci
Cosa vuoi fare? 1
digita l'intero da inserire nella pila
2
*****
***** MENU PRINCIPALE *****
*****
1> Push
2> Pop
3> Esci
Cosa vuoi fare? 3
Programma terminato
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

LISTE LINERARI N° 4

Si definisca una coda che contiene le informazioni relative alle pratiche da evadere nell'ufficio catastale di un comune di piccole dimensioni. Si implementino i sottoprogrammi che operano sulla coda. Infine, si sfruttino questi sottoprogrammi per costruire il programma che consente agli addetti del comune di aggiornare la coda delle pratiche.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

inizializzaCoda

Dati d'input: *c, puntatore alla **coda** (struct contenente **el** vettore elementi, primoEl, ultimoEl)

Dati in output:

Costanti: *MAX_NUM_EL* dimensione massima coda

estrai

Dati d'input: *c, puntatore alla coda

Dati in output: elem: elemento estratto oppure 0 se la coda è vuota

Costanti: *MAX_NUM_EL* dimensione massima coda

Variabili di lavoro: elem: elemento da estrarre

inserisci

Dati d'input: *c, elem, puntatore alla coda, elemento da inserire

Dati in output: 1 se inserimento avviene con successo, 0 viceversa

Costanti: *MAX_NUM_EL* dimensione massima coda

Variabili di lavoro:

main

Dati d'input: *

Dati in output:

Costanti: *MAX_NUM_EL* dimensione massima coda

Variabili di lavoro: C, numero, scelta, puntatore alla coda, numero da inserire o cancellare, scelta dell'operazione da effettuare tramite menù

3. Costruzione del modello

La seguente soluzione implementa una coda di interi. Estendere tale soluzione per risolvere il problema proposto nell'esercizio. Nella soluzione dell'esercizio sarebbe utile avere a disposizione un'ulteriore operazione che consenta di capire se la coda è piena oppure no. Provare a definire tale operazione mediante un sottoprogramma.

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_NUM_EL 5
typedef struct {
    int el[MAX_NUM_EL];
    int primoEl;
    int ultimoEl;} coda;
void inizializzaCoda(coda *c);
int estrai(coda *c);
int inserisci(int elem, coda *c);
int primoEl;
int ultimoEl;
int main(void)
{coda C; /* dichiarazione della coda */
```

```

int scelta, numero; /* variabili utilizzate per la gestione dei menu */
int ris; /* variabile utilizzata per memorizzare il risultato delle chiamate alle due funzioni di inserimento ed estrazione */
/* prepara la coda ad essere utilizzata */
inizializzaCoda(&C);
scelta=0;
while(scelta!=3){
printf("*****\n");
printf("***** MENU PRINCIPALE *****\n");
printf("*****\n\n");
printf("\t1) Inserisci\n");
printf("\t2) Estrai\n");
printf("\t3) Esci\n\n");
printf("Cosa vuoi fare? ");
scanf("%d",&scelta);
switch(scelta){
case 1:printf("digita l'intero da inserire nella coda\n");
scanf("%d", &numero);
if (numero != 0)
ris= inserisci(numero, &C);
if(ris == 0)
printf("Errore! La coda e' piena\n");
break;
case 2:numero = estrai(&C);
if (numero!=0)
printf("il numero prelevato dalla coda e' :%d\n", numero);
else printf("la coda e' vuota\n");
break;
case 3:printf("Programma terminato\n");
break;
default:printf("Devi inserire un numero compreso tra 1 e 3!");
break;}
printf("\n\n");
system("pause");
return(0);}
/* inserisce nella posizione ultimoEl della coda l'elemento il cui valore si trova in elem. Se la coda e` piena,
ritorna il valore zero che rappresenta un codice di errore */
int inserisci(int elem, coda *c){
if(c->ultimoEl < MAX_NUM_EL){
c->el[c->ultimoEl] = elem;
c->ultimoEl = c->ultimoEl + 1;
return (1); /* l'inserimento e` stato effettuato con successo*/
else return (0);}
/* estrae l'elemento che si trova in prima posizione (primoEl) nella coda. Se la coda e` vuota, ritorna il valore
0 */
int estrai(coda *c){
int elem;
if(c->primoEl!=c->ultimoEl){
/* c'e` almeno un elemento nella coda */
elem = c->el[c->primoEl];
c->primoEl = c->primoEl+1;}
else elem = 0; /* si noti che se 0 rappresenta un codice di errore, tale valore non deve mai apparire nella
coda come valore significativo */
return(elem);}
/* assegna un valore iniziale ai campi primoEl e ultimoEl della coda. Questo sottoprogramma deve essere

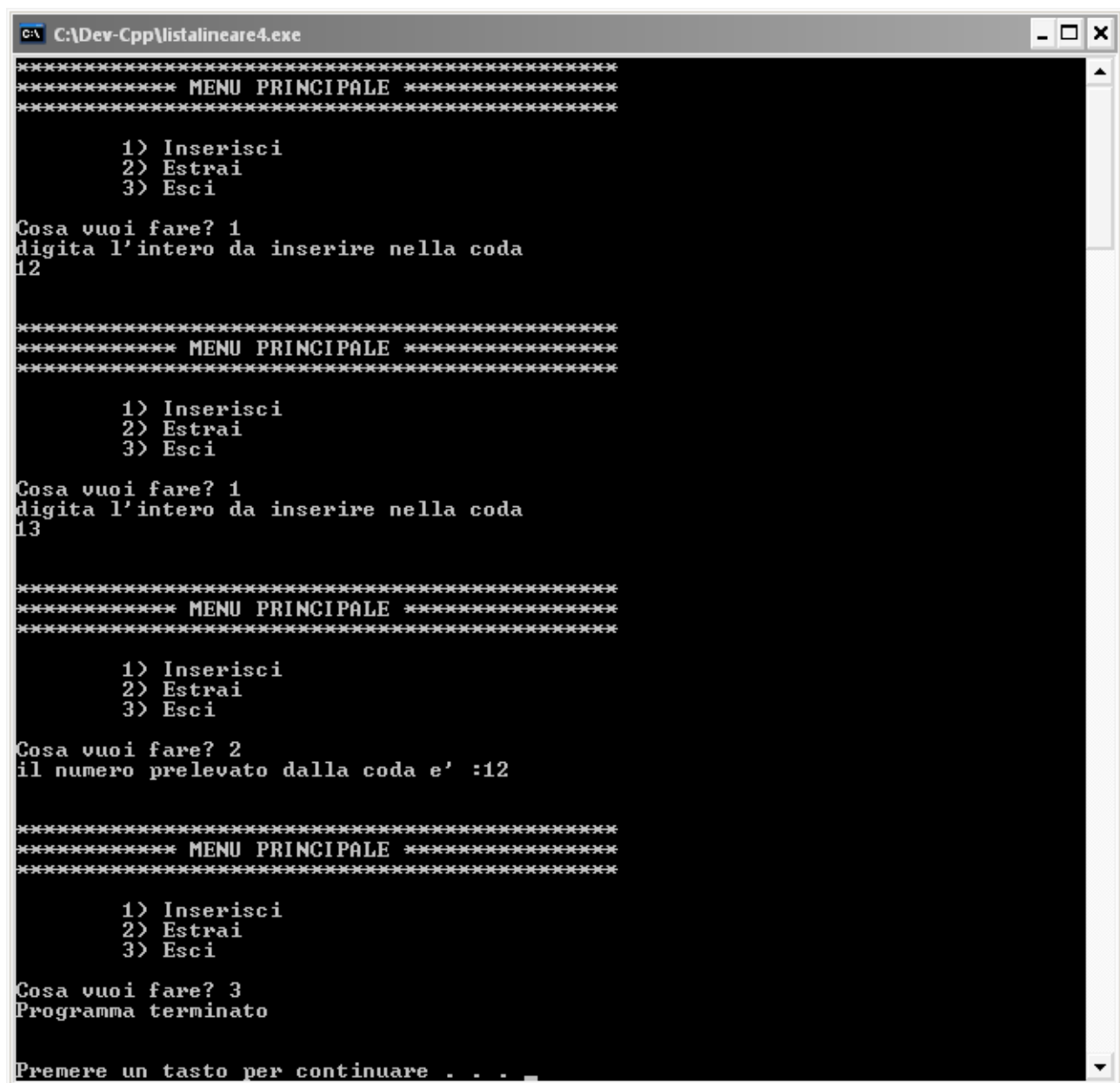
```

*sempre richiamato per preparare la coda ad essere utilizzata in modo corretto. */*

```
void inizializzaCoda(coda *c){
```

```
c->primoEl = 0;
```

```
c->ultimoEl = 0;}
```



```
C:\Dev-Cpp\listalineare4.exe
*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 1
digita l'intero da inserire nella coda
12

*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 1
digita l'intero da inserire nella coda
13

*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 2
il numero prelevato dalla coda e' :12

*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 3
Programma terminato

Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define MAX_NUM_EL 5
```

```
typedef struct {
```

```
int el[MAX_NUM_EL];
```

```
int primoEl;
```

```
int ultimoEl;} coda;
```

```
void inizializzaCoda(coda *c);
```

```
int estrai(coda *c);
```

```
int inserisci(int elem, coda *c);
```

```
int primoEl;
```

```
int ultimoEl;
```

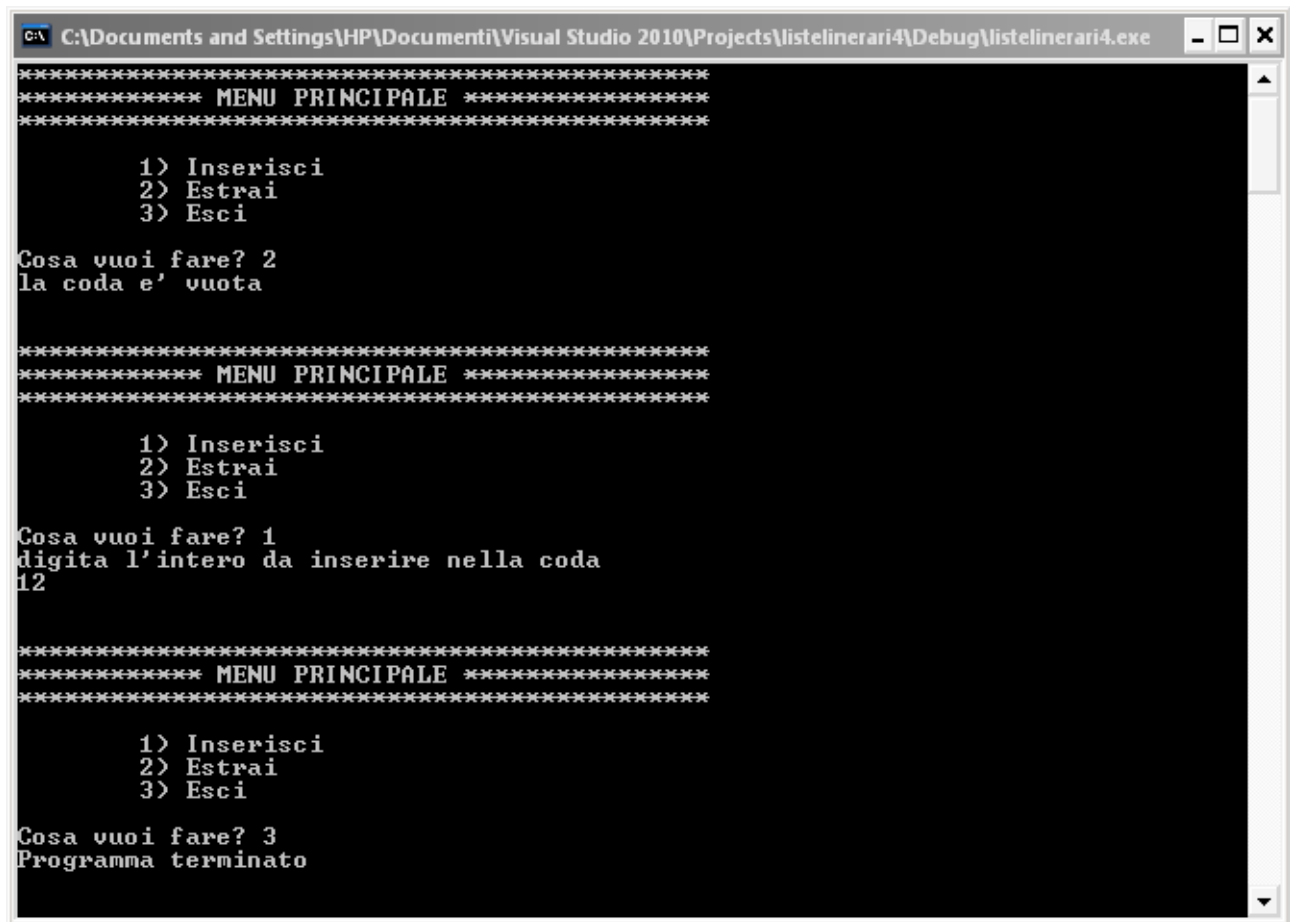
```
int main(void)
```

```

{coda C; /* dichiarazione della coda */
int scelta, numero; /* variabili utilizzate per la gestione dei menu */
int ris; /* variabile utilizzata per memorizzare il risultato delle chiamate alle due funzioni di inserimento ed estrazione */
inizializzaCoda(&C); /* prepara la coda ad essere utilizzata */
scelta=0;
while(scelta!=3)
{
printf("*****\n");
printf("***** MENU PRINCIPALE *****\n");
printf("*****\n\n");
printf("\t1) Inserisci\n");
printf("\t2) Estrai\n");
printf("\t3) Esci\n\n");
printf("Cosa vuoi fare? ");
scanf("%d",&scelta);
switch(scelta){
case 1:printf("digita l'intero da inserire nella coda\n");
scanf("%d", &numero);
if (numero != 0)
ris= inserisci(numero, &C);
if(ris == 0)
printf("Errore! La coda e' piena\n");
break;
case 2:numero = estrai(&C);
if (numero!=0)
printf("il numero prelevato dalla coda e' :%d\n", numero);
else printf("la coda e' vuota\n");
break;
case 3:printf("Programma terminato\n");
break;
default:printf("Devi inserire un numero compreso tra 1 e 3!");
break;}
printf("\n\n");
system("pause");
return(0);}
/* inserisce nella posizione ultimoEl della coda l'elemento il cui valore si trova in elem. Se la coda e' piena,
ritorna il valore zero che rappresenta un codice di errore */
int inserisci(int elem, coda *c){
if(c->ultimoEl < MAX_NUM_EL){
c->el[c->ultimoEl] = elem;
c->ultimoEl = c->ultimoEl + 1;
return (1); /* l'inserimento e' stato effettuato con successo*/
else return (0);}
/* estrae l'elemento che si trova in prima posizione (primoEl) nella coda. Se la coda e' vuota, ritorna il valore
0 */
int estrai(coda *c){
int elem;
if(c->primoEl!=c->ultimoEl){ /* c'e' almeno un elemento nella coda */
elem = c->el[c->primoEl];
c->primoEl = c->primoEl+1;}
else elem = 0; /* si noti che se 0 rappresenta un codice di errore, tale valore non deve mai apparire nella
coda come valore significativo */
return(elem);}
/* assegna un valore iniziale ai campi primoEl e ultimoEl della coda. Questo sottoprogramma deve essere

```

*sempre richiamato per preparare la coda ad essere utilizzata in modo corretto. */*
`void inizializzaCoda(coda *c){`
`c->primoEl = 0;`
`c->ultimoEl = 0;`
`}`



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\listelinerari4\Debug\listelinerari4.exe
*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 2
la coda e' vuota

*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 1
digita l'intero da inserire nella coda
12

*****
***** MENU PRINCIPALE *****
*****

1) Inserisci
2) Estrai
3) Esci

Cosa vuoi fare? 3
Programma terminato
```

5. Documentazione

6. Testing

TABELLE N° 1

Si vuole scrivere un programma C per la gestione di informazioni relative a studenti neolaureati in ingegneria. Tali informazioni sono contenute in un vettore in cui ogni elemento contiene i seguenti dati così strutturati:

- ✓ Cognome stringa contenuta in campo di esattamente 25 caratteri, può contenere spazi
- ✓ Nome stringa contenuta in campo di esattamente 25 caratteri, può contenere spazi
- ✓ corso di laurea può assumere i seguenti valori: Informatica, Elettronica, Civile, Meccanica, Elettrica, Gestionale
- ✓ voto di laurea intero, indica il voto di laurea
- ✓ lode intero, 1 indica la presenza di lode, 0 indica l'assenza
- ✓ numero anni di iscrizione intero, indica in quanti anni lo studente ha conseguito la laurea

Dopo aver definito una struttura dati (*struct studente*) e il vettore di *struct studente*, definire le seguenti funzioni: **RiempiVettore**, **StudentiMigliori**, **VisualizzaPerCdL**, **MediaCdL**, **main**.

1. Identificazione del sistema

Il sistema è di tipo elettrotecnico.

2. Analisi dei dati

RiempiVettore

Dati d'input: **v**, il vettore di *struct studente*, *dim* la sua dimensione

Dati di output:

Costanti: **N**, **M**, dimensione massima numero facoltà, dimensione massima per nome e cognome studenti

Variabili di lavoro: **i**, indice per spazzolare il vettore di *struct*.

StudentiMigliori

Dati d'input: **v**, il vettore di *struct studente*, *dim* la sua dimensione, **vsoglia** un voto soglia ed un numero anni soglia **annisoglia**

Dati di output: cognome, nome e voto di laurea degli studenti aventi un voto maggiore o uguale a **vsoglia** ed un numero di anni di iscrizione inferiore ad **annisoglia**. La funzione inoltre deve restituire il numero di studenti visualizzati, **count**

Costanti: **N**, **M**, dimensione massima numero facoltà, dimensione massima per nome e cognome studenti

Variabili di lavoro: **i**, indice per spazzolare il vettore di *struct*, **count** per il numero degli studenti visualizzati.

VisualizzaPerCdL

Dati d'input: **v**, il vettore di *struct studente*, *dim* la sua dimensione, **x**, *struct* di tipo **studente**.

Dati di output: **count1**, **count2**, numero di studenti aventi un voto pari a 110 o 110 e lode.

Costanti: **N**, **M**, dimensione massima numero facoltà, dimensione massima per nome e cognome studenti

Variabili di lavoro: **i**, indice per spazzolare il vettore di *struct*, **count1**, **count2**, numero studenti aventi un voto pari a 110 o 110 e lode.

MediaCdL

Dati d'input: **v**, il vettore di *struct studente*, *dim* la sua dimensione, **x**, *struct* di tipo **studente**, per corso di laurea da ricevere in input

Dati di output: **m/count**, la media dei voti del corso ricevuto in input

Costanti: **N**, **M**, dimensione massima numero facoltà, dimensione massima per nome e cognome studenti

Variabili di lavoro: **i**, indice per spazzolare il vettore di *struct*, **count** per contare il numero di studenti del corso, **m** per sommare i voti

Dati d'input: **v**, il vettore di *struct studente*, *dim* la sua dimensione, **x**, *struct* di tipo **studente**, per corso di laurea da ricevere in input

main

Dati d'input:

Dati di output:

Costanti: N ,M , dimensione massima numero facoltà, dimensione massima per nome e cognome studenti

Variabili di lavoro: *unsigned short scelta; struct studente elem;* scelta per il menù.

3. Costruzione del modello

Riempi Vettore : procedura che ha come parametri il vettore di *struct studente* e la sua dimensione: essa deve leggere le informazioni dalla tastiera inserendole all'interno del vettore di studenti

Studenti Migliori: funzione che accetta in ingresso un vettore di studenti, un voto soglia **vsoglia** ed un numero anni soglia **annisoglia**: essa deve visualizzare cognome, nome e voto di laurea degli studenti aventi un voto maggiore o uguale a *vsoglia* ed un numero di anni di iscrizione inferiore ad *annisoglia*. La funzione inoltre deve restituire il numero di studenti visualizzati.

VisualizzaPerCdL: funzione che accetta in ingresso un vettore di studenti e che per ogni corso di laurea previsto (Informatica, Elettronica, Civile, Meccanica, Elettrica, Gestionale) visualizzi il numero di studenti aventi un voto pari a 110 o 110 e lode.

MediaCdL: funzione che accetta in ingresso un vettore di studenti ed un corso di laurea: essa deve calcolare la media del voto di laurea degli studenti appartenenti a quel corso, restituendola al main (nota: ai fini del calcolo della media il 110 e lode si consideri pari a 110)

main: All'interno del main predisporre un opportuno menu che permetta di scegliere e far eseguire le diverse funzioni implementate.

4. Codifica

Dev-C++ 4.9.9.2

I SOLUZIONE: VETTORE Statico

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#define N 3
#define M 26
struct studente {
    char cognome[M], nome[M], cdL[M];
    unsigned short voto,lode,anni;};
struct studente archivio[N];
unsigned StudentiMigliori (struct studente v[], unsigned dim, unsigned vsoglia, unsigned
asoglia);
void RiempiVettore(struct studente v[], unsigned dim);
void VisualizzaPerCdL(struct studente v[], unsigned dim, struct studente x);
double MediaCdL(struct studente v[], unsigned dim, struct studente x);
int main(void){
    unsigned short scelta;
    struct studente elem;
    do {
        printf("\n1 Riempi Vettore ");
        printf("\n2 Visualizza Studenti Migliori ");
        printf("\n3 Visualizza Studenti per CdL ");
        printf("\n4 Calcolo Media ");
        printf("\n5 Fine ");
        printf("\nInserisci la tua scelta ---> ");
```

```

scanf("%u",&scelta);
fflush(stdin);
switch (scelta) {
    case 1: RiempiVettore(archivio,N);
        break;
    case 2: printf("\nInserisci il voto minimo ");
        scanf("%u",&elem.voto);
        fflush(stdin);
        printf("\nInserisci il numero di anni ");
        scanf("%u",&elem.anni);
        fflush(stdin);
        StudentiMigliori(archivio,N,elem.voto,elem.anni);
        break;
    case 3: printf("\nInserisci il cdl ");
        gets(elem.cdl);
        VisualizzaPerCdL(archivio,N,elem);
        break;
    case 4: printf("\nInserisci il cdl ");
        gets(elem.cdl);
        printf("\nLa Media e' %f",MediaCdL(archivio,N,elem));}
} while (scelta < 5);
system ("pause");
return(0);
}

void RiempiVettore(struct studente v[], unsigned dim){
    unsigned i;
    for (i=0; i<dim;i++) {
        printf("\nElemento %u ",i);
        printf("\nCognome ");
        gets(v[i].cognome);
        printf("\nnome ");
        gets(v[i].nome);
        printf("\nCdL ");
        gets(v[i].cdl);
        printf("\nVoto di Laurea ");
        scanf("%u",&v[i].voto);
        fflush(stdin);
        printf("\nLode (1-si, 0-no) ");
        scanf("%u",&v[i].lode);
        fflush(stdin);
        printf("\nAnni di Studio ");
        scanf("%u",&v[i].anni);
        fflush(stdin);
    }
}

unsigned StudentiMigliori (struct studente v[], unsigned dim, unsigned vsoglia, unsigned
asoglia){
    unsigned i,count=0;

    for (i=0;i<dim; i++) {
        if (v[i].voto>=vsoglia &&v[i].anni<asoglia){
            printf("\nCognome %s ",v[i].cognome);
            printf("\nNome %s ",v[i].nome);
            printf("\nVoto %u ",v[i].voto);
            count++; }
    }
}

```

```

    }
    return (count);
}
void VisualizzaPerCdL(struct studente v[], unsigned dim, struct studente x){
    unsigned i, count1=0, count2=0;
    for (i=0;i<dim; i++) {
        if (!strcmp(v[i].cdl,x.cdl) && v[i].voto==110)
            if (v[i].lode==0) count1++;
            else count2++;}
    printf("\nIl numero di studenti con 110 e' %u ",count1);
    printf("\nIl numero di studenti con 110 e lode e' %u ",count2);
}
double MediaCdL(struct studente v[], unsigned dim, struct studente x)
{
    unsigned i,count=0;
    double m=0.0;

    for (i=0;i<dim; i++) {
        if (!strcmp(v[i].cdl,x.cdl)) {
            m+=v[i].voto;
            count++;}
    }
    return (m/count);}

```

```
C:\Documents and Settings\HP\Documenti\tabelle\tabelle1.exe

1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> 1

Elemento 0
Cognome re
nome anna
CdL informatica
Voto di Laurea 110
Lode <1-si, 0-no> 0
Anni di Studio 5

Elemento 1
Cognome rossi
nome mario
CdL informatica
Voto di Laurea 80
Lode <1-si, 0-no> 0
Anni di Studio 5

Elemento 2
Cognome roi
nome alba
CdL informatica
Voto di Laurea 89
Lode <1-si, 0-no> 0
Anni di Studio 5

1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> 2

Inserisci il voto minimo 80
Inserisci il numero di anni 5

1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> 3

Inserisci il cdl informatica

Il numero di studenti con 110 e' 0
Il numero di studenti con 110 e lode e' 0
1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> 5
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#define N 3
#define M 26
struct studente {
    char cognome[M], nome[M], cdl[M];
    unsigned short voto,lode,anni;};
struct studente archivio[N];
unsigned StudentiMigliori (struct studente v[], unsigned dim, unsigned vsoglia, unsigned
asoglia);
void RiempiVettore(struct studente v[], unsigned dim);
void VisualizzaPerCdL(struct studente v[], unsigned dim, struct studente x);
double MediaCdL(struct studente v[], unsigned dim, struct studente x);
int main(void){
    unsigned short scelta;
    struct studente elem;
    do {
        printf("\n1 Riempi Vettore ");
        printf("\n2 Visualizza Studenti Migliori ");
        printf("\n3 Visualizza Studenti per CdL ");
        printf("\n4 Calcolo Media ");
        printf("\n5 Fine ");
        printf("\nInserisci la tua scelta ---> ");
        scanf("%u",&scelta);
        fflush(stdin);
        switch (scelta) {
            case 1: RiempiVettore(archivio,N);
                break;
            case 2: printf("\nInserisci il voto minimo ");
                scanf("%u",&elem.voto);
                fflush(stdin);
                printf("\nInserisci il numero di anni ");
                scanf("%u",&elem.anni);
                fflush(stdin);
                StudentiMigliori(archivio,N,elem.voto,elem.anni);
                break;
            case 3: printf("\nInserisci il cdl ");
                gets(elem.cdl);
                VisualizzaPerCdL(archivio,N,elem);
                break;
            case 4: printf("\nInserisci il cdl ");
                gets(elem.cdl);
                printf("\nLa Media e' %f",MediaCdL(archivio,N,elem));}
        } while (scelta < 5);
        system ("pause");
        return(0);
    }
void RiempiVettore(struct studente v[], unsigned dim){
    unsigned i;
    for (i=0; i<dim;i++) {
        printf("\nElemento %u ",i);
        printf("\nCognome ");
```

```

    gets(v[i].cognome);
    printf("\nnome ");
    gets(v[i].nome);
    printf("\nCdL ");
    gets(v[i].cdl);
    printf("\nVoto di Laurea ");
    scanf("%u",&v[i].voto);
    fflush(stdin);
    printf("\nLode (1-si, 0-no) ");
    scanf("%u",&v[i].lode);
    fflush(stdin);
    printf("\nAnni di Studio ");
    scanf("%u",&v[i].anni);
    fflush(stdin);
}
}
unsigned StudentiMigliori (struct studente v[], unsigned dim, unsigned vsoglia, unsigned
asoglia){
    unsigned i,count=0;

    for (i=0;i<dim; i++) {
        if (v[i].voto>=vsoglia &&v[i].anni<asoglia){
            printf("\nCognome %s ",v[i].cognome);
            printf("\nNome %s ",v[i].nome);
            printf("\nVoto %u ",v[i].voto);
            count++; }
    }
    return (count);
}
void VisualizzaPerCdL(struct studente v[], unsigned dim, struct studente x){
    unsigned i, count1=0, count2=0;
    for (i=0;i<dim; i++) {
        if (!strcmp(v[i].cdl,x.cdl) && v[i].voto==110)
            if (v[i].lode==0) count1++;
            else count2++;}
    printf("\nIl numero di studenti con 110 e' %u ",count1);
    printf("\nIl numero di studenti con 110 e lode e' %u ",count2);
}
double MediaCdL(struct studente v[], unsigned dim, struct studente x)
{
    unsigned i,count=0;
    double m=0.0;

    for (i=0;i<dim; i++) {
        if (!strcmp(v[i].cdl,x.cdl)) {
            m+=v[i].voto;
            count++;}
    }
    return (m/count);}

```

```
C:\Documents and Settings\HP\documenti\visual studio 2010\Projects\tabelle1\Debug\tabelle1.exe
1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> 1

Elemento 0
Cognome re
nome anna
CdL chimica
Voto di Laurea 90
Lode <1-si, 0-no> 0
Anni di Studio 4

Elemento 1
Cognome rossi
nome mario
CdL chimica
Voto di Laurea 100
Lode <1-si, 0-no> 0
Anni di Studio 4

Elemento 2
Cognome rosi
nome alma
CdL matematica
Voto di Laurea 90
Lode <1-si, 0-no> 0
Anni di Studio 4

1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> 4

Inserisci il cdl chimica

La Media e' 95.000000
1 Riempi Vettore
2 Visualizza Studenti Migliori
3 Visualizza Studenti per CdL
4 Calcolo Media
5 Fine
Inserisci la tua scelta ---> _
```


Dev-C++ 4.9.9.2

II SOLUZIONE : VETTORE Dinamico

```
#include<stdio.h>
#include<string.h>
#include<alloc.h>
unsigned n; /*la dimensione del vettore */

struct studente {
    char cognome[26], nome[26], cdl[10];
    unsigned short voto,lode,anni;
};

struct studente *archivio;
```

TUTTE LE FUNZIONI IDENTICHE a PRIMA

```
Int main(void){

    unsigned short scelta;
    struct studente elem;

    printf("\nInserisci la dimensione del vettore ---> ");
    scanf("%u",&n);
    fflush(stdin);
    archivio=(struct studente *)malloc(n*sizeof(struct studente));

    IL RESTO del MAIN IDENTICO a PRIMA, a patto di cambiare N con n
    free (archivio);
}
```

5. Documentazione

6. Testing

TABELLE N° 2

Si definisca un sottoprogramma che effettua una ricerca per numero di matricola in un array che contiene informazioni relative agli studenti di un corso. Se l'elemento cercato si trova nell'array, il sottoprogramma restituisce al chiamante l'**indirizzo** della cella corrispondente. Si supponga che l'array su cui si effettua la ricerca sia **ordinato**. Si definiscano le strutture dati necessarie per lo sviluppo del sottoprogramma

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

ricerca

Dati d'input: *a, puntatori a **archivioSt** (struct contenente l'array stud di tipo **studente** (a sua volta struct contenente: matricola, nome, cognome dello studente) e numSt), matr: numero di matricola

Dati di output: a.stud[i]: record dello studente trovato, oppure NULL se non è presente nell'archivio.

Costanti : *MAX_STR 20, MAX_NUM_EL 10*

Variabili di lavoro: i , indice per spazzolare l'array **stud**

3. Costruzione del modello

4. Codifica: sottoprogramma ricerca

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include<stdlib.h>
#define MAX_STR 20
#define MAX_NUM_EL 10
typedef char stringa[MAX_STR];
typedef struct {
    int matricola;
    stringa nome;
    stringa cognome;
} studente;
typedef struct {
    studente stud[MAX_NUM_EL];
    int numSt;
} archivioSt;

studente * ricerca(archivioSt *a, int matr)
{
    int i;
    for (i=0; i<a->numSt && matr>a->stud[i].matricola; i++);
    if (matr == a->stud[i].matricola)
        return &a->stud[i];
    else return NULL;
}
```

5. Documentazione

6. Testing

TABELLE N° 3

Si definisca:

Un tipo TipoArchivio con campi NumDate di tipo intero e SeqDate array di 100 elementi di tipo Data.

Un tipo VettoreDiPuntatori come array di 100 elementi di tipo puntatore a elementi di tipo Data (giorno, mese anno). Si scriva un main che utilizza una variabile A1 di tipo TipoArchivio, una variabile VP di tipo VettoreDiPuntatori ed una variabile DataOdierna di tipo Data. Scrivere la porzione di un main che, per ogni elemento della Sequenza contenuta nell'archivio A1, confronta l'elemento del vettore con la data odierna. Se l'elemento del vettore precede la data odierna il main assegna un puntatore del vettore VP all'elemento della sequenza appena analizzato. Eseguire l'assegnamento del vettore prima internamente al main e poi tramite procedura

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

LeggiData

Dati d'input:

Dati in output:

Costanti: N, numero massimo di date

Variabili di lavoro: d di tipo Data, struct contenente giorno, mese, anno.

StampaData

Dati d'input: d di tipo Data, struct contenente giorno, mese, anno.

Dati in output: stampa le date inserite se minori della data odierna

Costanti: N, numero massimo di date

Variabili di lavoro:

Precede

Dati d'input: d1, d2, di tipo data

Dati in output: confronta due date

Costanti: N, numero massimo di date

Variabili di lavoro:

main

Dati d'input:

Dati in output:

Costanti: N, numero massimo di date

Variabili di lavoro: A1, struct di tipo **TipoArchivio** (struct contenente vettore di date di tipo **data** e NumDate), VP, tabella di tipo VettoreDiPuntatori, DataOdierna di tipo Data, i, j indici per spazzolare la tabella.

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define N 100
```

```
typedef struct {
```

```
int giorno;
```

```
int mese;
```

```
int anno;
```

```
} Data;
```

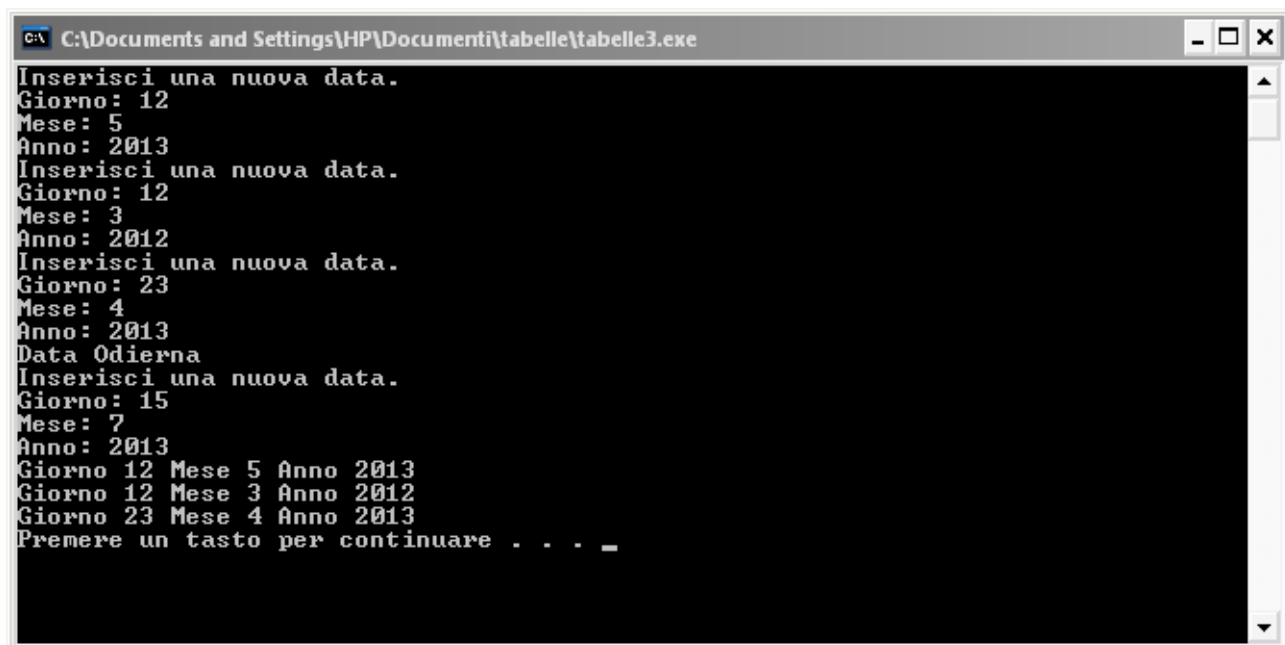
```
typedef struct {
```

```
Data SeqDate[N];
```

```

int NumDate;
} TipoArchivio;
typedef Data* VettoreDiPuntatori[N];
typedef enum {falso, vero} boolean;
Data LeggiData();
void StampaData(Data d);
boolean Precede(Data d1, Data d2);
int main(void){
    TipoArchivio A1;
    VettoreDiPuntatori VP;
    Data DataOdierna;
    int i, j;
    A1.NumDate=0;
    for(i=0;i<3;i++){
        A1.SeqDate[i]=LeggiData();
        A1.NumDate++;}
    printf("Data Odierna\n");
    DataOdierna = LeggiData();
    j=0;
    for(i=0;i<A1.NumDate;i++)
        if(Precede(A1.SeqDate[i], DataOdierna) == vero){
            VP[j] = &A1.SeqDate[i];
            j++;}
    for (i=0;i<j;i++)
        StampaData(*(VP[i]));
    system("pause");
    return(0);}
Data LeggiData(){
    Data d;
    printf("Inserisci una nuova data.\nGiorno: ");
    scanf("%d", &d.giorno);
    printf("Mese: ");
    scanf("%d", &d.mese);
    printf("Anno: ");
    scanf("%d", &d.anno);
    return d;}
void StampaData(Data d){
    printf("Giorno %d ", d.giorno);
    printf("Mese %d ", d.mese);
    printf("Anno %d\n", d.anno);}
boolean Precede(Data d1, Data d2){
    if(d1.anno<d2.anno)
        return vero;
    else if(d1.anno==d2.anno)
        if(d1.mese<d2.mese)
            return vero;
        else if(d1.mese==d2.mese)
            if(d1.giorno<=d2.giorno)
                return vero;
            else return falso;}

```



```
C:\Documents and Settings\HP\Documenti\tabelle\tabelle3.exe
Inserisci una nuova data.
Giorno: 12
Mese: 5
Anno: 2013
Inserisci una nuova data.
Giorno: 12
Mese: 3
Anno: 2012
Inserisci una nuova data.
Giorno: 23
Mese: 4
Anno: 2013
Data Odierna
Inserisci una nuova data.
Giorno: 15
Mese: 7
Anno: 2013
Giorno 12 Mese 5 Anno 2013
Giorno 12 Mese 3 Anno 2012
Giorno 23 Mese 4 Anno 2013
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

```
#include <stdio.h>
#include<stdlib.h>
typedef struct {
int giorno;
int mese;
int anno;
} Data;
typedef struct {
Data SeqDate[100];
int NumDate;
} TipoArchivio;
typedef Data* VettoreDiPuntatori[100];
typedef enum {falso, vero} boolean;
Data LeggiData();
void StampaData(Data d);
boolean Precede(Data d1, Data d2);

int main(void)
{
TipoArchivio A1;
VettoreDiPuntatori VP;
Data DataOdierna;
int i, j;
A1.NumDate=0;
for(i=0;i<3;i++)
{
A1.SeqDate[i]=LeggiData();
A1.NumDate++;
}
printf("Data Odierna\n");
DataOdierna = LeggiData();
j=0;
for(i=0;i<A1.NumDate;i++)
if(Precede(A1.SeqDate[i], DataOdierna) == vero)
```

```

{
    VP[j] = &A1.SeqDate[i];
    j++;
}
for (i=0;i<j;i++)
    StampaData(*(VP[i]));
system("pause");
return(0);}

Data LeggiData(){
    Data d;
    printf("Inserisci una nuova data.\nGiorno: ");
    scanf("%d", &d.giorno);
    printf("Mese: ");
    scanf("%d", &d.mese);
    printf("Anno: ");
    scanf("%d", &d.anno);
    return d;}

void StampaData(Data d){
    printf("Giorno %d ", d.giorno);
    printf("Mese %d ", d.mese);
    printf("Anno %d\n", d.anno);}

boolean Precede(Data d1, Data d2){
    if(d1.anno<d2.anno)
        return vero;
    else if(d1.anno==d2.anno)
        if(d1.mese<d2.mese)
            return vero;
        else if(d1.mese==d2.mese)
            if(d1.giorno<=d2.giorno)
                return vero;
            else return falso;}

```

```

c:\documents and settings\hp\documenti\visual studio 2010\Projects\tabelle3\Debug\tabelle3.exe
Inserisci una nuova data.
Giorno: 23
Mese: 4
Anno: 2013
Inserisci una nuova data.
Giorno: 24
Mese: 5
Anno: 2013
Inserisci una nuova data.
Giorno: 18
Mese: 8
Anno: 2013
Data Odierna
Inserisci una nuova data.
Giorno: 15
Mese: 7
Anno: 2013
Giorno 23 Mese 4 Anno 2013
Giorno 24 Mese 5 Anno 2013
Premere un tasto per continuare . . . _

```

5. Documentazione

6. Testing

TABELLE N° 4

Dopo aver caricato in modo dinamico una tabella di studenti, conenete i relativi nomi, le età e un vettore di voti, stampare la media dei voti di ogni studente con il relativo nome.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

carica

Dati d'input: classe, puntatore a studente (struct contenente nome, età e vettore con tre voti), n numero di studenti

Dati in output:

Costanti: N, numero massimo di studenti, MATERIE, numero massimo di voti

Variabili di lavoro: i, j, contatori di cicli for

stampamedie

Dati d'input: classe, puntatore a studente (struct contenente nome, età e vettore con tre voti), n numero di studenti

Dati in output: m, media dei voti di ogni studente, classe[i].nome, nomi degli studenti

Costanti: N, numero massimo di studenti, MATERIE, numero massimo di voti

Variabili di lavoro: m, media dei voti, i, j contatori cicli for, sommaVoti, per calcolare la somma dei voti degli studenti

main

Dati d'input:

Dati in output:

Costanti: N, numero massimo di studenti, MATERIE, numero massimo di voti

Variabili di lavoro: classe, puntatore a studente (struct contenente nome, età e vettore con tre voti), n numero di studenti

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define MATERIE 3
```

```
#define N 20
```

```
typedef struct { char nome[N];
```

```
    int eta;
```

```
    int voto[MATERIE];
```

```
} studente;
```

```
void carica (studente *classe, int n);
```

```
void stampamedie (studente *classe, int n);
```

```
int main(void)
```

```
{
```

```
    studente *classe; //vettore di struct con puntatori
```

```
    int n;
```

```
    printf("inserisci numero studenti, max 20\t");
```

```
    scanf("%d",&n);
```

```
    classe=(studente *)malloc(n*sizeof(studente)); //alloco vettore di n elementi di studenti
```

```
    carica(classe,n);
```

```
    stampamedie(classe,n);
```

```
    system("pause");
```

```
    return(0);
```

```
}
```

```

void carica (studente *classe, int n){
int j,i;
for(i=0;i<n; i++) {
printf("inserisci nome, eta del %d studente", i+1);
scanf("%s\t %d", classe[i].nome, &classe[i].eta);
printf("inserisci i voti delle tre materie del %d studente", i+1);
for(j=0; j<=2; j++)
scanf("%d", &classe[i].voto[j]);
}
}

void stampamedie(studente *classe, int n){
float m;
int i,j,sommaVoti;
for(i=0;i<n; i++)
{
sommaVoti=0;
for(j=0; j<=2; j++)
sommaVoti+=classe[i].voto[j];
m = (float)sommaVoti/MATERIE;
printf("media di %s: %f\n", classe[i].nome, m);
}
}

```

```

C:\Dev-Cpp\tabelle4.exe
inserisci numero studenti, max 20      3
inserisci nome, eta del 1 studentere
23
inserisci i voti delle tre materie del 1 studente23
24
25
inserisci nome, eta del 2 studentetre
24
inserisci i voti delle tre materie del 2 studente25
26
27
inserisci nome, eta del 3 studenteioipa
24
inserisci i voti delle tre materie del 3 studente30
23
24
media di re: 24.000000
media di tre: 26.000000
media di iopa: 25.666666
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

#include <stdio.h>
#include<stdlib.h>
#define MATERIE 3
#define N 20
typedef struct { char nome[N];
int eta;
int voto[MATERIE];
} studente;
void carica (studente *classe, int n);
void stampamedie (studente *classe, int n);
int main(void)

```



```

{
    studente *classe; //vettore di struct con puntatori
    int n;
    printf("inserisci numero studenti, max 20\t");
    scanf("%d",&n);
    classe=(studente *)malloc(n*sizeof(studente)); //alloca vettore di n elementi di studenti
    carica(classe,n);
    stampamedie(classe,n);
    system("pause");
    return(0);
}

void carica (studente *classe, int n){
    int j,i;
    for(i=0;i<n; i++) {
        printf("inserisci nome, eta del %d studente", i+1);
        scanf("%s\t %d", classe[i].nome, &classe[i].eta);
        printf("inserisci i voti delle tre materie del %d studente", i+1);
        for(j=0; j<=2; j++)
            scanf("%d", &classe[i].voto[j]);
    }
}

void stampamedie(studente *classe, int n){
    float m;
    int i,j,sommaVoti;
    for(i=0;i<n; i++)
    {
        sommaVoti=0;
        for(j=0; j<=2; j++)
            sommaVoti+=classe[i].voto[j];
        m = (float)sommaVoti/MATERIE;
        printf("media di %s: %f\n", classe[i].nome, m);
    }
}

```

```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\tab1all\Debug\tab1all.exe
inserisci numero studenti, max 20      5
inserisci nome, eta del 1 studenterossi
23
inserisci i voti delle tre materie del 1 studente23
24
25
inserisci nome, eta del 2 studentere
21
inserisci i voti delle tre materie del 2 studente24
25
30
inserisci nome, eta del 3 studenteera
24
inserisci i voti delle tre materie del 3 studente21
21
21
inserisci nome, eta del 4 studentewer
23
inserisci i voti delle tre materie del 4 studente23
24
30
inserisci nome, eta del 5 studenterae
34
inserisci i voti delle tre materie del 5 studente23
23
23
media di rossi: 24.000000
media di re: 26.333334
media di era: 21.000000
media di wer: 25.666666
media di rae: 23.000000
Premere un tasto per continuare . . . _
```

5. Documentazione

6. Testing

FILE N° 1 (FILE DI TESTO)

Si scriva un programma che legge da un file di testo una sequenza di stringhe e le stampa sullo schermo.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

leggi

Dati d'input: *in, str, puntatore al file, stringa da leggere.

Dati in output: lettura file in.txt

Costanti: N, dimensione massima della stringa da leggere.

Variabili di lavoro: letti, valore di ritorno della fscanf

main

Dati d'input:

Dati in output:

Costanti: N, dimensione massima della stringa da leggere.

Variabili di lavoro: *in, str[n], puntatore al file, stringa da leggere

3. Costruzione del modello

La presente soluzione assume che il file contenente la sequenza di stringhe esiste già. È possibile crearlo utilizzando un qualsiasi editor di testo. Il nome del file da creare è in.txt.

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define N 10
```

```
void leggi (FILE *in, char str[N]);
```

```
int main(void){
```

```
FILE *in; char str[N];
```

```
in=fopen("in.txt", "r");
```

```
if(in==NULL)
```

```
printf("Impossibile aprire il file\n");
```

```
else
```

```
leggi(in, str);
```

```
printf("\n");
```

```
system ("pause");
```

```
return(0);}
```

```
void leggi (FILE *in, char str[N])
```

```
{int letti;
```

```
letti=fscanf(in, "%s", str);
```

```
/* il seguente ciclo termina se si verifica una delle seguenti situazioni:viene raggiunta la fine di in.txt, si verifica un errore di lettura da in.txt*/
```

```
while(letti>0){
```

```
printf("%s\n", str);
```

```
letti=fscanf(in, "%s", str);}
```

```
/*si verifica se l'uscita dal ciclo sia avvenuta per la terminazione del file in.txt oppure per il verificarsi di un errore*/
```

```
if(ferror(in)==1)
```

```
printf("Errore di i/o");
```

```
if(fclose(in)==EOF)
```

```
printf("Errore di chiusura file");}
```



```
C:\Dev-Cpp\file1.exe
1
2
3
4
5
6
7
8
23
Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
#define N 10
void leggi (FILE *in, char str[N]);
int main(void){
FILE *in; char str[N];
in=fopen("in.txt", "r");
if(in==NULL)
printf("Impossibile aprire il file\n");
else
leggi(in, str);
printf("\n");
system("pause");
return(0);}
void leggi (FILE *in, char str[N])
{int letti;
letti=fscanf(in, "%s", str);
/* il seguente ciclo termina se si verifica una delle seguenti situazioni:viene raggiunta la fine di in.txt, si
verifica un errore di lettura da in.txt*/
while(letti>0){
printf("%s\n", str);
letti=fscanf(in, "%s", str);}
/*si verifica se l'uscita dal ciclo sia avvenuta per la terminazione del file in.txt oppure per il verificarsi di un
errore*/
if(ferror(in)==1)
printf("Errore di i/o");
if(fclose(in)==EOF)
printf("Errore di chiusura file");}
```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file1\Debug\file1.exe
2
3
4
5
6
7
8
88
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

FILE N° 2

Si scriva un programma che acquisisce una sequenza di interi da standard input e la salva su un file in modalità testo.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

scrivi

Dati d'input: *p, puntatore al file su cui scrivere.

Dati in output: val

Variabili di lavoro: val, valore da scrivere sul file con fprintf

main

Dati d'input

Dati in output: val

Variabili di lavoro: : *p, puntatore al file su cui scrivere

3. Costruzione del modello

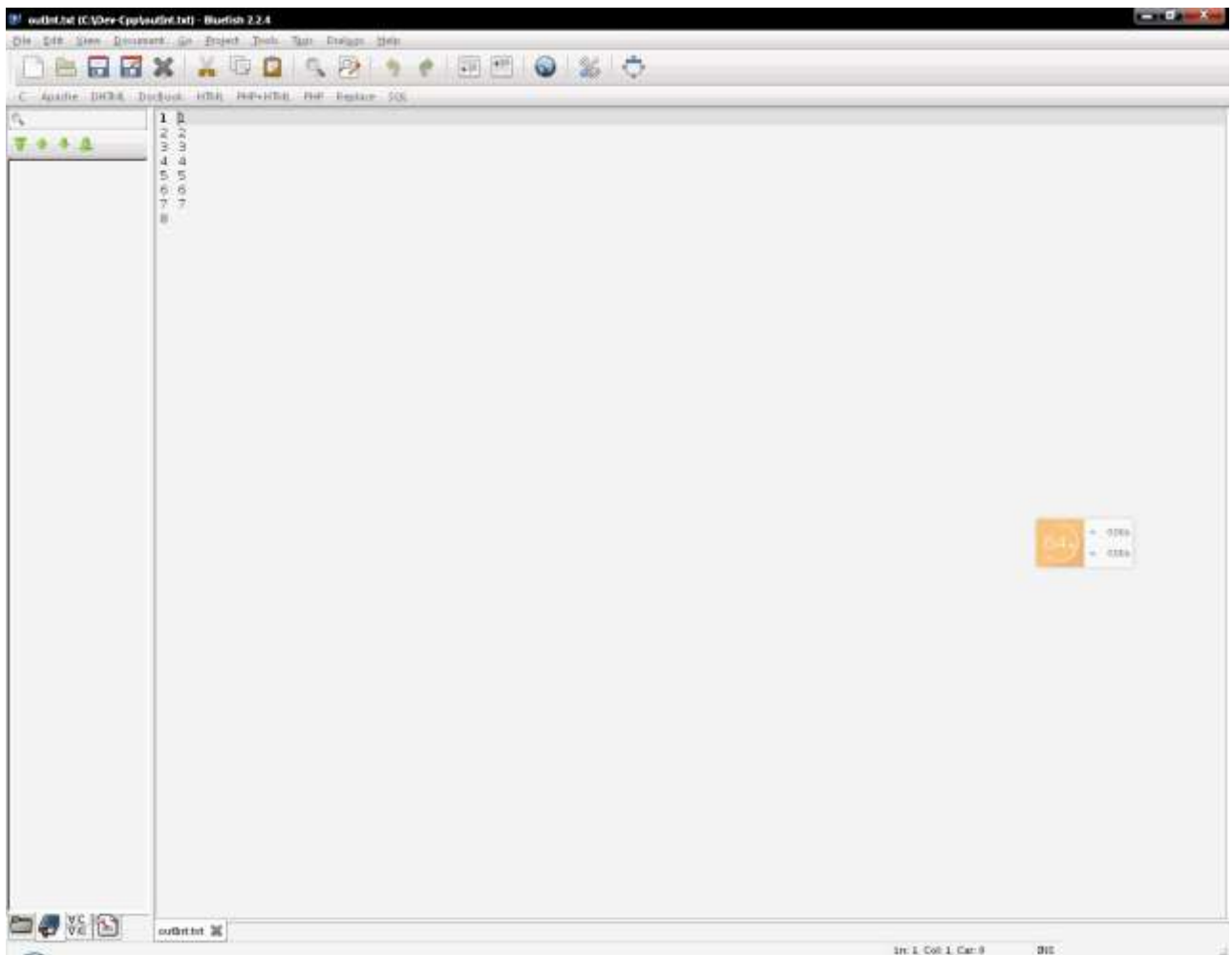
4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
void scrivi(FILE *p);
int main(void)
{
    FILE *p;
    int scritti;
    p=fopen("outInt.txt", "w");
    if(p==NULL)
        printf("Impossibile aprire il file");
    else
    {
        scrivi(p);
        if(fclose(p)==EOF)
            printf("Errore di chiusura file");
        system("pause");
        return(0);
    }
}
void scrivi(FILE *p){
    int val;
    printf("Inserisci una sequenza di interi terminata dal numero 999\n");
    scanf("%d", &val);
    while(val!=999)
    {
        fprintf(p, "%d\n", val);
        scanf("%d", &val);
    }
}
```



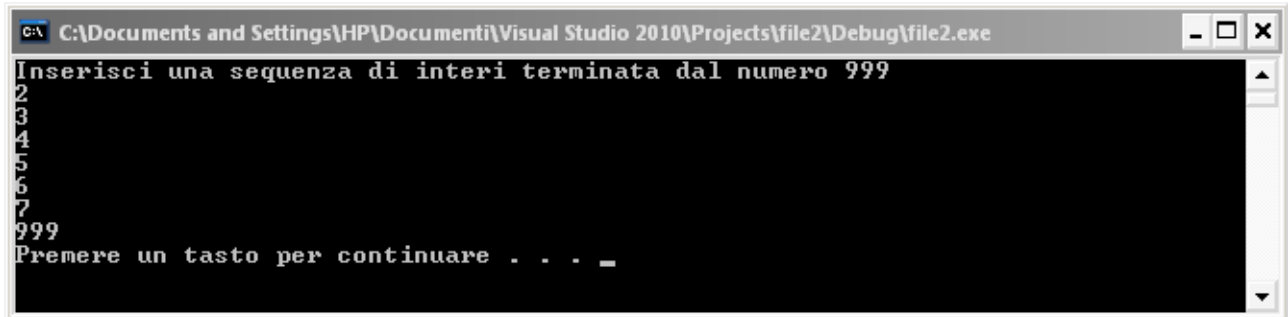
```
C:\Dev-Cpp\file2.exe
Inserisci una sequenza di interi terminata dal numero 999
1
2
3
4
5
6
7
999_
```



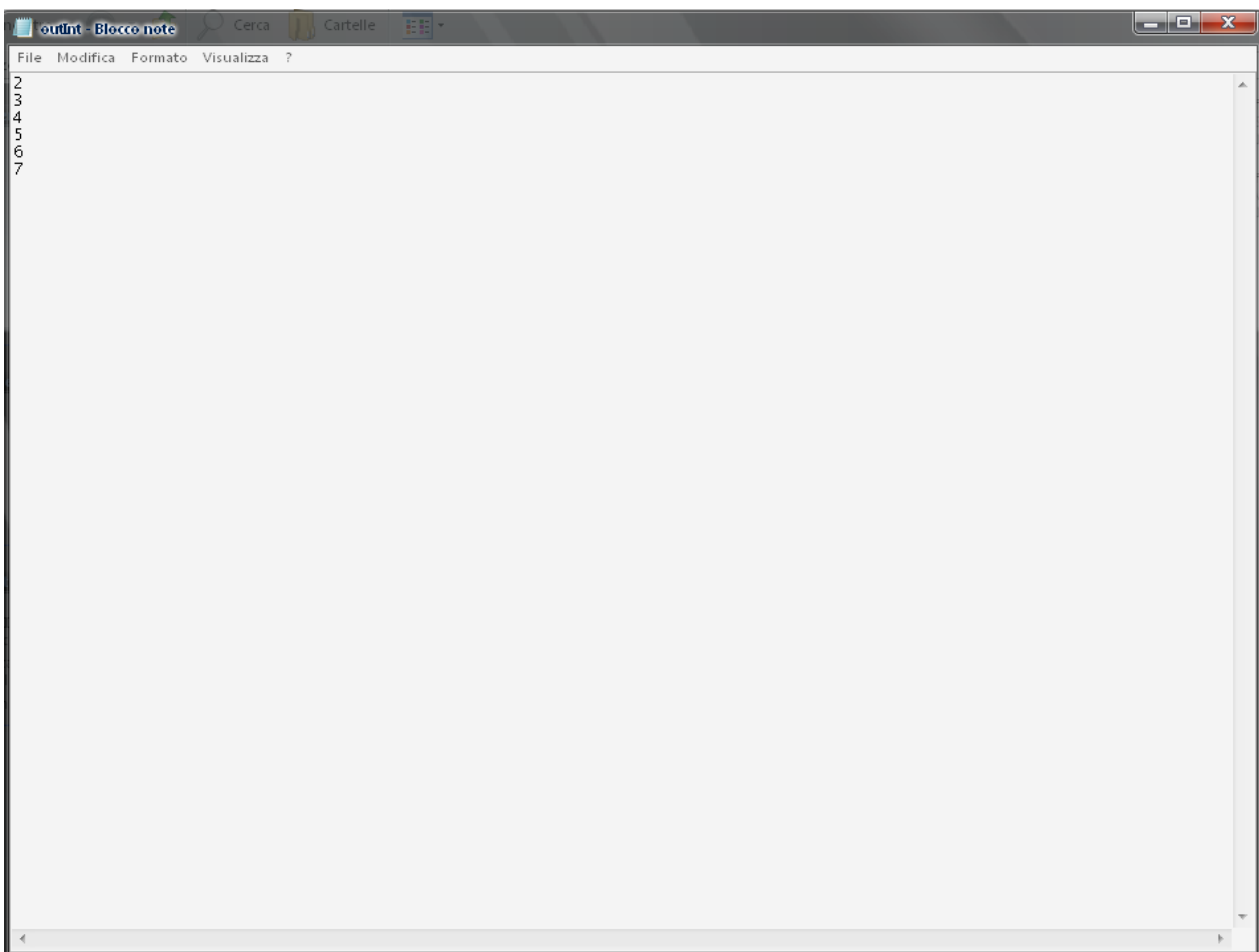
Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
void scrivi(FILE *p);
int main(void)
{
    FILE *p;
    int scritti;
    p=fopen("outInt.txt","w");
    if(p==NULL)
        printf("Impossibile aprire il file");
    else
    {scrivi(p);
    if(fclose(p)==EOF)
        printf("Errore di chiusura file");
    system("pause");
```

```
return(0);}}  
void scrivi(FILE *p){  
int val;  
printf("Inserisci una sequenza di interi terminata dal numero 999\n");  
scanf("%d", &val);  
while(val!=999)  
{fprintf(p, "%d\n", val);  
scanf("%d", &val);}}
```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file2\Debug\file2.exe  
Inserisci una sequenza di interi terminata dal numero 999  
2  
3  
4  
5  
6  
7  
999  
Premere un tasto per continuare . . . _
```



```
outint - Blocco note  
File Modifica Formato Visualizza ?  
2  
3  
4  
5  
6  
7  
999
```

5. Documentazione

6. Testing

FILE N° 3

Scrivete un programma che legge da un file di testo una sequenza di interi, la ordina in modo crescente e salva su file la sequenza risultante sovrascrivendo il vecchio contenuto.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

ordina

Dati d'input: tmp, numElem, vettore da ordinare, elementi del vettore

Dati in output: vettore ordinato

Costanti: MAX_NUM dimensione del vettore da leggere.

Variabili di lavoro: i, cont, indice dei cicli for, temp variabile temporanea per scambio.

main

Dati d'input:

Dati in output: scrittura vettore ordinato su file in.txt (sovrascrivo)

Costanti: MAX_NUM dimensione del vettore da leggere.

Variabili di lavoro: tmp, letti, i, scritti, vettore in cui memorizzare i numeri letti dal file, numero letto dal file, indice ciclo while, flag che indica la presenza di numeri da scrivere sul file

3. Costruzione del modello

La presente soluzione assume che il file contenente la sequenza di interi esiste già. È possibile crearlo utilizzando un qualsiasi editor di testo. Il nome del file da creare è in.txt. Estendete la soluzione in modo da consentire all'utente di indicare il nome del file contenente la sequenza di interi.

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define MAX_NUM 10
```

```
void ordina(int a[], int dim);
```

```
int main(void){
```

```
FILE *arch;
```

```
int tmp[10];
```

```
int letti, scritti;
```

```
int i, numElem;
```

```
arch=fopen("in.txt", "r");
```

```
if(arch==NULL)
```

```
printf("Impossibile aprire il file");
```

```
else{
```

```
letti = 1;
```

```
i=0;
```

```
while(letti>0 && i<MAX_NUM){
```

```
letti=fscanf(arch, "%d", &tmp[i]);
```

```
i++;}
```

```
/* numElem viene usata per tenere traccia del numero di elementi significativi contenuti in tmp */
```

```
numElem = i;
```

```
/* ordino l'array */
```

```
ordina(tmp, numElem);
```

```
/* adesso tutto è pronto per riscrivere gli elementi nel file attenzione però! Il file deve essere riaperto nella modalità retta */
```

```
if(fclose(arch)==EOF)
```

```
printf("Errore di chiusura file");
```



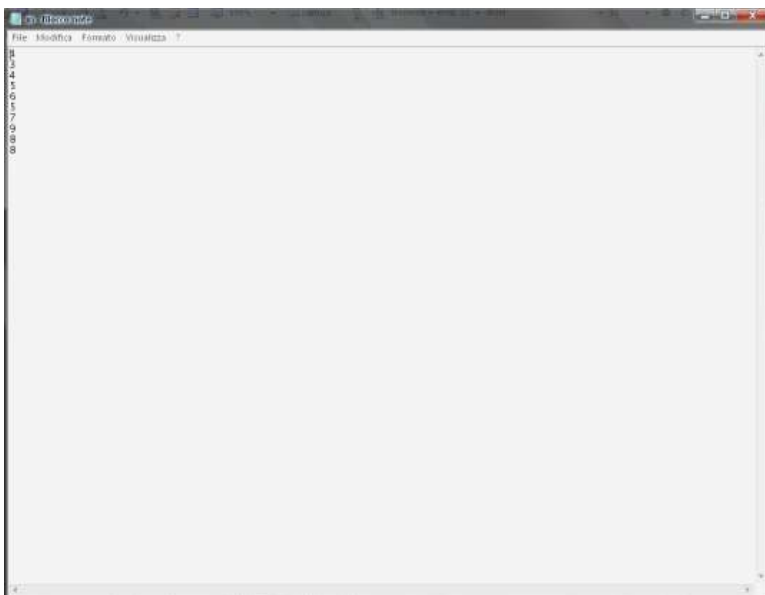
```

arch=fopen("in.txt", "w");
if(arch==NULL)
printf("Impossibile aprire il file");
else{
scritti = 1;
i=0;
while(scritti>0 && i<numElem){
letti=fprintf(arch,"%d ",tmp[i]);
i++;}
if(ferror(arch)==1)
printf("Errore di i/o");
if(fclose(arch)==EOF)
printf("Errore di chiusura file");}}
system("pause");
return(0);}

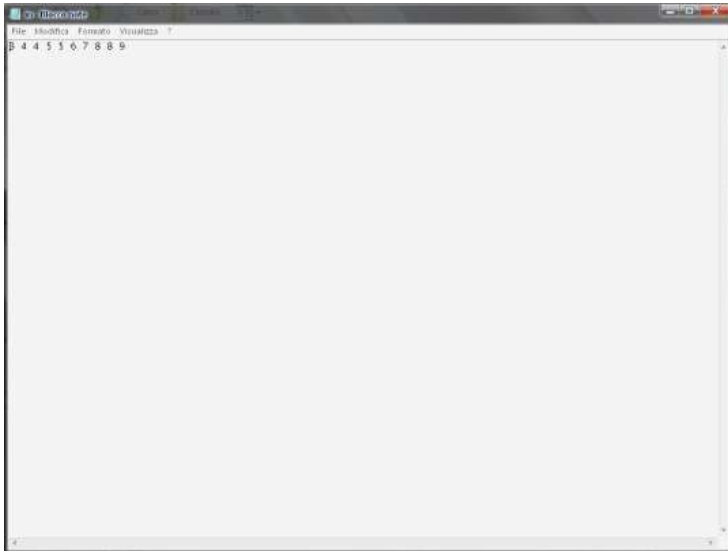
void ordina(int a[], int dim){
int cont, i, tmp;
for (cont=0;cont<dim-1;cont++){
/* E' il ciclo più interno. Vengono confrontati l'elemento in posizione i-esima e quello in posizione i+1-esima.
Se quest'ultimo e' più piccolo viene effettuato lo scambio di posizione */
for (i=0; i<dim-1; i++){
if (a[i]>a[i+1]){
/* Scambio fra gli elementi. Viene utilizzata la variabile temporanea temp */
tmp = a[i];
a[i] = a[i+1];
a[i+1] = tmp;
}
}
}
}
}

```

File in.txt prima dell'esecuzione del programma:



File in.txt dopo l'esecuzione del programma:



Microsoft Visual C++

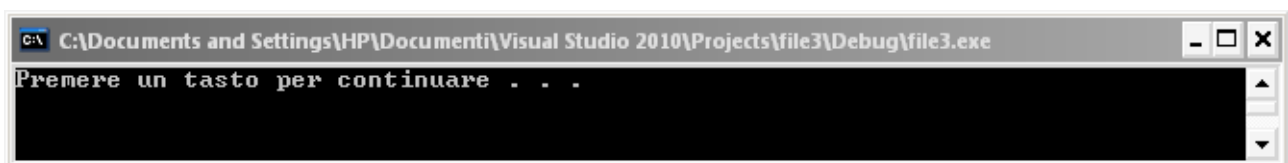
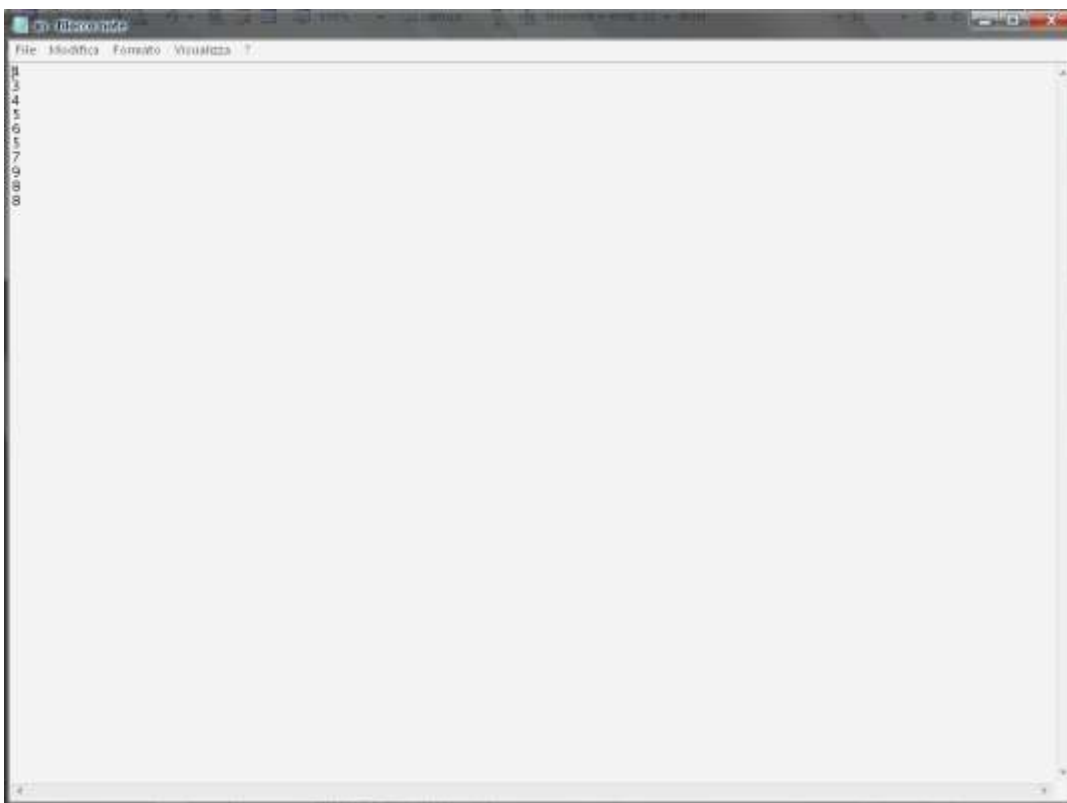
```
#include <stdio.h>
#include <stdlib.h>
#define MAX_NUM 10
void ordina(int a[], int dim);
int main(void){
    FILE *arch;
    int tmp[10];
    int letti, scritti;
    int i, numElem;
    arch=fopen("in.txt", "r");
    if(arch==NULL)
        printf("Impossibile aprire il file");
    else{
        letti = 1;
        i=0;
        while(letti>0 && i<MAX_NUM){
            letti=fscanf(arch, "%d", &tmp[i]);
            i++;}
        /* numElem viene usata per tenere traccia del numero di elementi significativi contenuti in tmp */
        numElem = i;
        /* ordino l'array */
        ordina(tmp, numElem);
        /* adesso tutto è pronto per riscrivere gli elementi nel file attenzione però! Il file deve essere riaperto nella
        modalità retta */
        if(fclose(arch)==EOF)
            printf("Errore di chiusura file");
        arch=fopen("in.txt", "w");
        if(arch==NULL)
            printf("Impossibile aprire il file");
        else{
            scritti = 1;
            i=0;
            while(scritti>0 && i<numElem){
                letti=fprintf(arch, "%d ", tmp[i]);
```

```

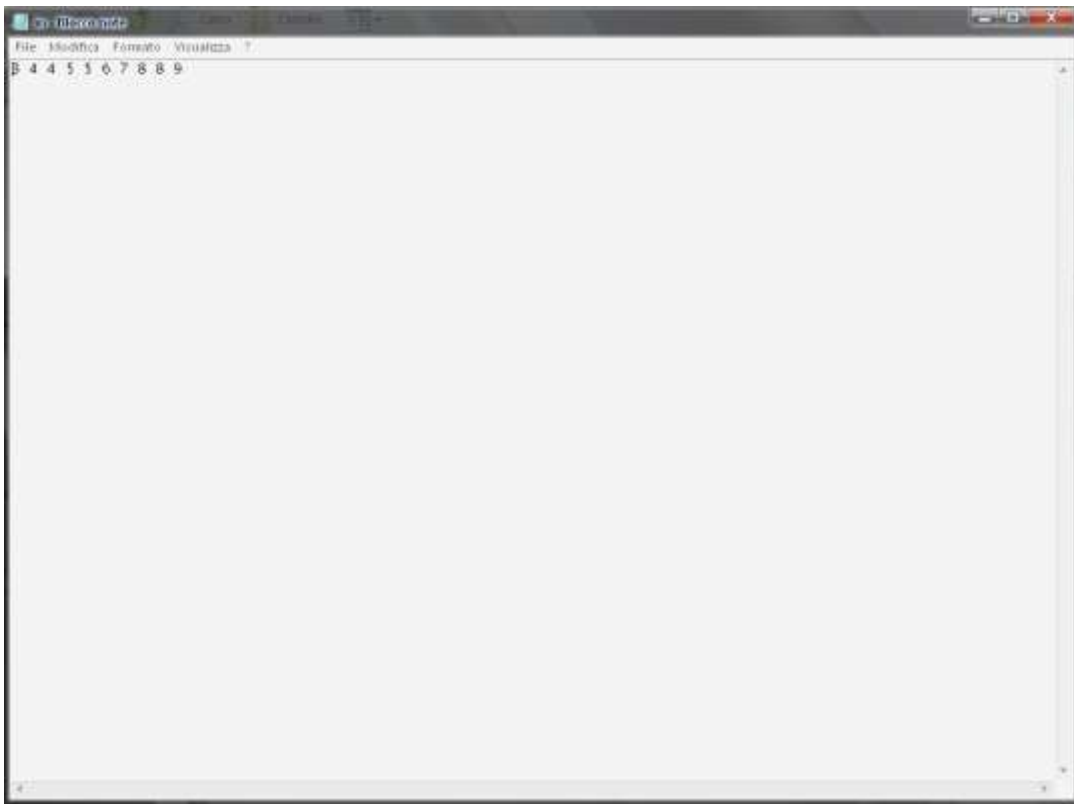
i++;}
if(ferror(arch)==1)
printf("Errore di i/o");
if(fclose(arch)==EOF)
printf("Errore di chiusura file");}}
system("pause");
return(0);}
void ordina(int a[], int dim){
int cont, i, tmp;
for (cont=0;cont<dim-1;cont++){
/* E' il ciclo più interno. Vengono confrontati l'elemento in posizione i-esima e quello in posizione i+1-esima.
Se quest'ultimo e' più piccolo viene effettuato lo scambio di posizione */
for (i=0; i<dim-1; i++){
if (a[i]>a[i+1]){
/* Scambio fra gli elementi. Viene utilizzata la variabile temporanea temp */
tmp = a[i];
a[i] = a[i+1];
a[i+1] = tmp;}}}}

```

File in.txt prima dell'esecuzione del programma:



File in.txt dopo l'esecuzione del programma:



5. Documentazione

6. Testing

FILE N° 4

Si implementi un programma che crea un file in modalità testo e che inserisce nel file valori interi (voti) acquisiti dallo standard input, fino ad un massimo di 20. (voti.txt) Il secondo legge i dati dal file e ne calcola la media, stampandola a video ed "appendendola" ad un altro file di testo aperto in modalità "append" (media.txt).

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

main

Dati d'input:

Dati in output: creazione file voti.txt

Costanti: NOME_FILE_VOTI, MAX_VOTI, nome del file e numero massimo di voti

Variabili di lavoro: : *file, votoEsame, numeroEsami, nElementi, puntatore al file su cui scrivere, per memorizzare un singolo voto da leggere o scrivere, per contare i voti già inseriti, per memorizzare il numero di elementi letti o scritti su file ad ogni operazione di IO.

3. Costruzione del modello

La soluzione si compone di due programmi distinti. Il primo inserisce i voti in un file, il secondo legge questi voti e calcola la media. Il primo programma è implementato in due versioni. La prima versione aggiunge ad ogni esecuzione nuovi valori al file dei voti. La seconda versione ad ogni esecuzione distrugge il contenuto del file dei voti prima di inserire nuovi valori.

4. Codifica

Programma per la scrittura dei voti sul file - prima versione

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define NOME_FILE_VOTI "voti.txt"
```

```
#define MAX_VOTI 20
```

```
/*Programma che permette all'utente di inserire all'interno di un file di testo un massimo di 20 voti. Ad ogni esecuzione il programma verifica il numero di voti già presenti nel file e permette all'utente di aggiungerne di nuovi in numero tale da non superare la soglia di 20*/
```

```
int main(void){
```

```
FILE *file;
```

```
int votoEsame; /* per memorizzare un singolo voto da leggere o scrivere */
```

```
int numeroEsami; /* per contare i voti gia' inseriti */
```

```
int nElementi; /* per memorizzare il numero di elementi letti o scritti su file ad ogni operazione di IO */
```

```
/***** conteggio dei voti gia' inseriti *****/
```

```
numeroEsami=0;
```

```
file=fopen(NOME_FILE_VOTI, "r");
```

```
if(file==NULL)
```

```
printf("Impossibile aprire il file: nessun voto inserito");
```

```
else{
```

```
printf("Voti gia' inseriti: \n");
```

```
nElementi=fscanf(file, "%d", &votoEsame);
```

```
while(nElementi>0){
```

```
numeroEsami++;
```

```
printf("%d ", votoEsame);
```

```
nElementi=fscanf(file, "%d", &votoEsame);}
```

```
if(fclose(file)!=0 || ferror(file)==1){
```

```
printf("Errore di IO: impossibile continuare");
```

```

return 0; /* questa istruzione provoca laterminazione del programma */}
/***** inserimento nuovi voti *****/
file=fopen(NOME_FILE_VOTI,"a");
if(file==NULL)
printf("Impossibile aprire il file");
else{
printf("\nInserimento nuovi voti. ");
printf("Immettere un numero negativo per terminare\n");
nElementi=1;
votoEsame=0;
numeroEsami++;
while(numeroEsami<=MAX_VOTI && votoEsame>=0 && nElementi>0){
printf("%d: ",numeroEsami);
scanf("%d",&votoEsame);
if(votoEsame>=0)
nElementi=fprintf(file,"%d\n",votoEsame);
numeroEsami++;}
if(fclose(file)!=0 || ferror(file)==1)
printf("Errore di IO");
system("pause");
return(0);}

```

```

C:\Dev-Cpp\file4.exe
Voti gia' inseriti:
4 5 6 7
Inserimento nuovi voti. Immettere un numero negativo per terminare
5: 6
6: 7
7: 7
8: 8
9: 9
10: -1
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

#include <stdio.h>
#include <stdlib.h>
#define NOME_FILE_VOTI "voti.txt"
#define MAX_VOTI 20
/*Programma che permette all'utente di inserire all'interno di un file di testo un massimo di 20 voti. Ad ogni
esecuzione il programma verifica il numero di voti già presenti nel file e permette all'utente di aggiungerne di
nuovi in numero tale da non superare la soglia di 20*/
int main(void){
FILE *file;
int votoEsame; /* per memorizzare un singolo voto da leggere o scrivere */
int numeroEsami; /* per contare i voti gia' inseriti */
int nElementi; /* per memorizzare il numero di elementi letti o scritti su file ad ogni operazione di IO */
/***** conteggio dei voti gia' inseriti *****/
numeroEsami=0;
file=fopen(NOME_FILE_VOTI, "r");
if(file==NULL)
printf("Impossibile aprire il file: nessun voto inserito");
else{
printf("Voti gia' inseriti: \n");
nElementi=fscanf(file,"%d",&votoEsame);
while(nElementi>0){

```

```

numeroEsami++;
printf("%d ", votoEsame);
nElementi=fscanf(file, "%d",&votoEsame);}
if(fclose(file)!=0 || ferror(file)==1){
printf("Errore di IO: impossibile continuare");
return 0; /* questa istruzione provoca la terminazione del programma */}
/***** inserimento nuovi voti *****/
file=fopen(NOME_FILE_VOTI,"a");
if(file==NULL)
printf("Impossibile aprire il file");
else{
printf("\nInserimento nuovi voti. ");
printf("Immettere un numero negativo per terminare\n");
nElementi=1;
votoEsame=0;
numeroEsami++;
while(numeroEsami<=MAX_VOTI && votoEsame>=0 && nElementi>0){
printf("%d: ", numeroEsami);
scanf("%d",&votoEsame);
if(votoEsame>=0)
nElementi=fprintf(file, "%d\n", votoEsame);
numeroEsami++;}
if(fclose(file)!=0 || ferror(file)==1)
printf("Errore di IO");
system("pause");
return(0);}}

```

```

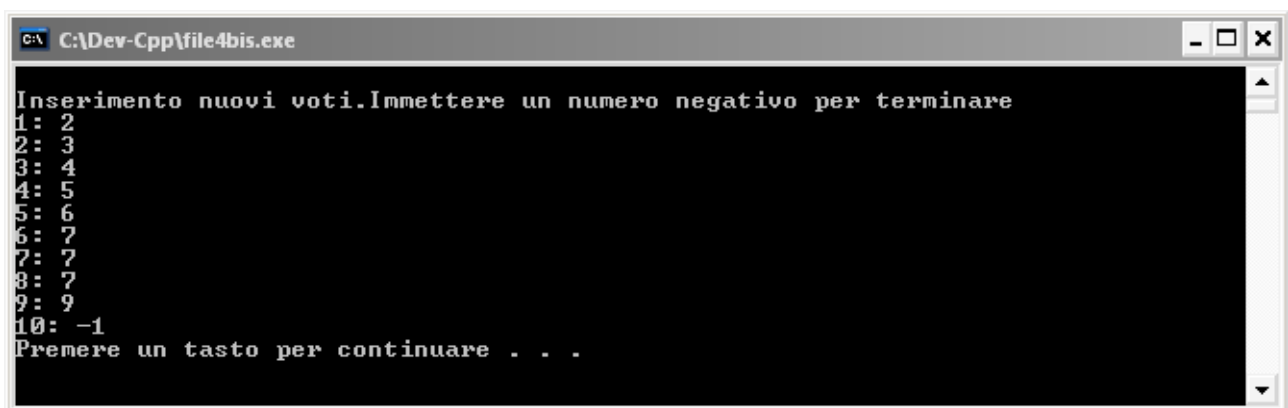
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file4one\Debug\file4one.exe
Voti gia' inseriti:
2 3 4 5 6 7 8 9 10
Inserimento nuovi voti. Immettere un numero negativo per terminare
10: 7
11: 8
12: 9
13: -1
Premere un tasto per continuare . . . _

```

Programma per la scrittura dei voti sul file - seconda versione

Dev-C++ 4.9.9.2


```
#include <stdio.h>
#include <stdlib.h>
#define NOME_FILE_VOTI "voti.txt"
#define MAX_VOTI 20
/* Programma che permette all'utente di inserire all'interno di un file di testo un massimo di 20 voti. Ad ogni
esecuzione il programma cancella tutti i voti eventualmente già presenti nel file (cioè viene cancellato il
risultato delle esecuzioni precedenti).*/
int main(void){
FILE *file;
int votoEsame; /* per memorizzare un singolo voto da leggere o scrivere */
int numeroEsami; /* per contare i voti già inseriti */
int nElementi; /* per memorizzare il numero di elementi letti o scritti su file ad ogni operazione di IO */
file=fopen(NOME_FILE_VOTI,"w");
if(file==NULL)
printf("Impossibile aprire il file");
else{
printf("\nInserimento nuovi voti.");
printf("Immettere un numero negativo per terminare\n");
nElementi=1;
votoEsame=0;
numeroEsami=1;
while(numeroEsami<=MAX_VOTI && votoEsame>=0 && nElementi>0){
printf("%d: ",numeroEsami);
scanf("%d",&votoEsame);
if(votoEsame>=0)
nElementi=fprintf(file,"%d\n",votoEsame);
numeroEsami++;}
if(fclose(file)!=0 || ferror(file)==1)
printf("Errore di IO");
system("pause");
return(0);
}
}
```



```
C:\Dev-Cpp\file4bis.exe
Inserimento nuovi voti.Immettere un numero negativo per terminare
1: 2
2: 3
3: 4
4: 5
5: 6
6: 7
7: 7
8: 7
9: 9
10: -1
Premere un tasto per continuare . . .
```


Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
#define NOME_FILE_VOTI "voti.txt"
#define MAX_VOTI 20
/* Programma che permette all'utente di inserire all'interno di un file di testo un massimo di 20 voti. Ad ogni
esecuzione il programma cancella tutti i voti eventualmente già presenti nel file (cioè viene cancellato il
risultato delle esecuzioni precedenti).*/
int main(void){
FILE *file;
int votoEsame; /* per memorizzare un singolo voto da leggere o scrivere */
int numeroEsami; /* per contare i voti già inseriti */
int nElementi; /* per memorizzare il numerodi elementi letti o scritti su file ad ogni operazione di IO */
file=fopen(NOME_FILE_VOTI,"w");
if(file==NULL)
printf("Impossibile aprire il file");
else{
printf("\nInserimento nuovi voti( max 20).");
printf("Immettere un numero negativo per terminare\n");
nElementi=1;
votoEsame=0;
numeroEsami=1;
while(numeroEsami<=MAX_VOTI && votoEsame>=0 && nElementi>0){
printf("%d: ",numeroEsami);
scanf("%d",&votoEsame);
if(votoEsame>=0)
nElementi=fprintf(file,"%d\n",votoEsame);
numeroEsami++;}
if(fclose(file)!=0 || ferror(file)==1)
printf("Errore di IO");
system ("pause");
return(0);}}
```



```
C:\documents and settings\hpb\documenti\visual studio 2010\Projects\file4two\Debug\file4two.exe
Inserimento nuovi voti.Immettere un numero negativo per terminare
1: 1
2: 2
3: 1
4: 2
5: 3
6: 4
7: 5
8: 6
9: 7
10: 8
11: 9
12: 6
13: 5
14: 7
15: 8
16: 4
17: 3
18: 5
19: 6
20: 7
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

FILE N° 5

Si implementi un programma che legge i dati dal file creato nell'esercizio precedente (voti.txt) e ne calcola la media stampandola a video ed "appendendola" ad un altro file di testo aperto in modalità "append" (media.txt).

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

main

Dati d'input:

Dati in output: creazione file media.txt

Costanti: NOME_FILE_IN, NOME_FILE_OUT, nome del file letto, nome del file creato contenente la media .

Variabili di lavoro: *fileIn, *fileOut, votoEsame, numeroEsami, nEI, totale, numero Esami, puntatore al file da cui leggere, puntatore al file su cui scrivere, per memorizzare un voto da leggere, per contare i voti già inseriti, per le operazioni di lettura/scrittura, memorizza la somma di tutti i voti letti, per il conteggio del numero di voti presenti nel file.

3. Costruzione del modello

4. Codifica


Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include<stdlib.h>
#define NOME_FILE_IN "voti.txt"
#define NOME_FILE_OUT "media.txt"
/*Programma che legge una sequenza di voti da un file di testo e ne calcola la media. La media calcolata
viene inserita alla fine di un secondo file di testo.*/
int main(void)
{FILE *fileIn, *fileOut;
float media; /* per memorizzare la media */
int votoEsame; /* per memorizzare un voto letto da file */
int nEI; /* per le operazioni di lettura/scrittura*/
int totale; /* memorizza la somma di tutti i voti letti */
int numeroEsami; /* per il conteggio del numero di voti presenti nel file*/
numeroEsami=0;
totale=0;
/***** lettura dei voti, calcolo del totale e del loro numero *****/
fileIn=fopen(NOME_FILE_IN, "r");
if(fileIn==NULL)
printf("Impossibile aprire il file di input");
else{
nEI=fscanf(fileIn,"%d",&votoEsame);
while(nEI>0 && numeroEsami < 20){
totale=totale+votoEsame;
numeroEsami++;
printf("%d ",votoEsame);
nEI=fscanf(fileIn,"%d",&votoEsame);}
if(ferror(fileIn)==1)
printf("Errore in lettura");
else if(numeroEsami!=0){
/***** calcolo media e scrittura su file *****/
media=(float)totale/(float)numeroEsami;
```

```

printf("\nMEDIA: %f\n",media);
fileOut=fopen(NOME_FILE_OUT,"a");
if(fileOut==NULL)
printf("Impossibile aprire il file di output");
else{
fprintf(fileOut,"MEDIA: %f\n",media);
if(ferror(fileOut)==1)
printf("Errore in scrittura");
if(fclose(fileOut)!=0)
printf("Errore durante la chiusura di un file");}}
if(fclose(fileIn)!=0)
printf("Errore durante la chiusura di un file");
system("pause");
return(0);}}

```



Microsoft Visual C++

```

#include <stdio.h>
#include<stdlib.h>
#define NOME_FILE_IN "voti.txt"
#define NOME_FILE_OUT "media.txt"
/*Programma che legge una sequenza di voti da un file di testo e ne calcola la media. La media calcolata
viene inserita alla fine di un secondo file di testo.*/
int main(void)
{FILE *fileIn, *fileOut;
float media; /* per memorizzare la media */
int votoEsame; /* per memorizzare un voto letto da file */
int nEl; /* per le operazioni di lettura/scrittura*/
int totale; /* memorizza la somma di tutti i voti letti */
int numeroEsami; /* per il conteggio del numero di voti presenti nel file*/
numeroEsami=0;
totale=0;
/***** lettura dei voti, calcolo del totale e del loro numero *****/
fileIn=fopen(NOME_FILE_IN, "r");
if(fileIn==NULL)
printf("Impossibile aprire il file di input");
else{
nEl=fscanf(fileIn,"%d",&votoEsame);
while(nEl>0 && numeroEsami < 20){
totale=totale+votoEsame;
numeroEsami++;
printf("%d ",votoEsame);
nEl=fscanf(fileIn,"%d",&votoEsame);}
if(ferror(fileIn)==1)
printf("Errore in lettura");
else if(numeroEsami!=0){
/***** calcolo media e scrittura su file *****/
media=(float)totale/(float)numeroEsami;
printf("\nMEDIA: %f\n",media);
}
}
}

```

```

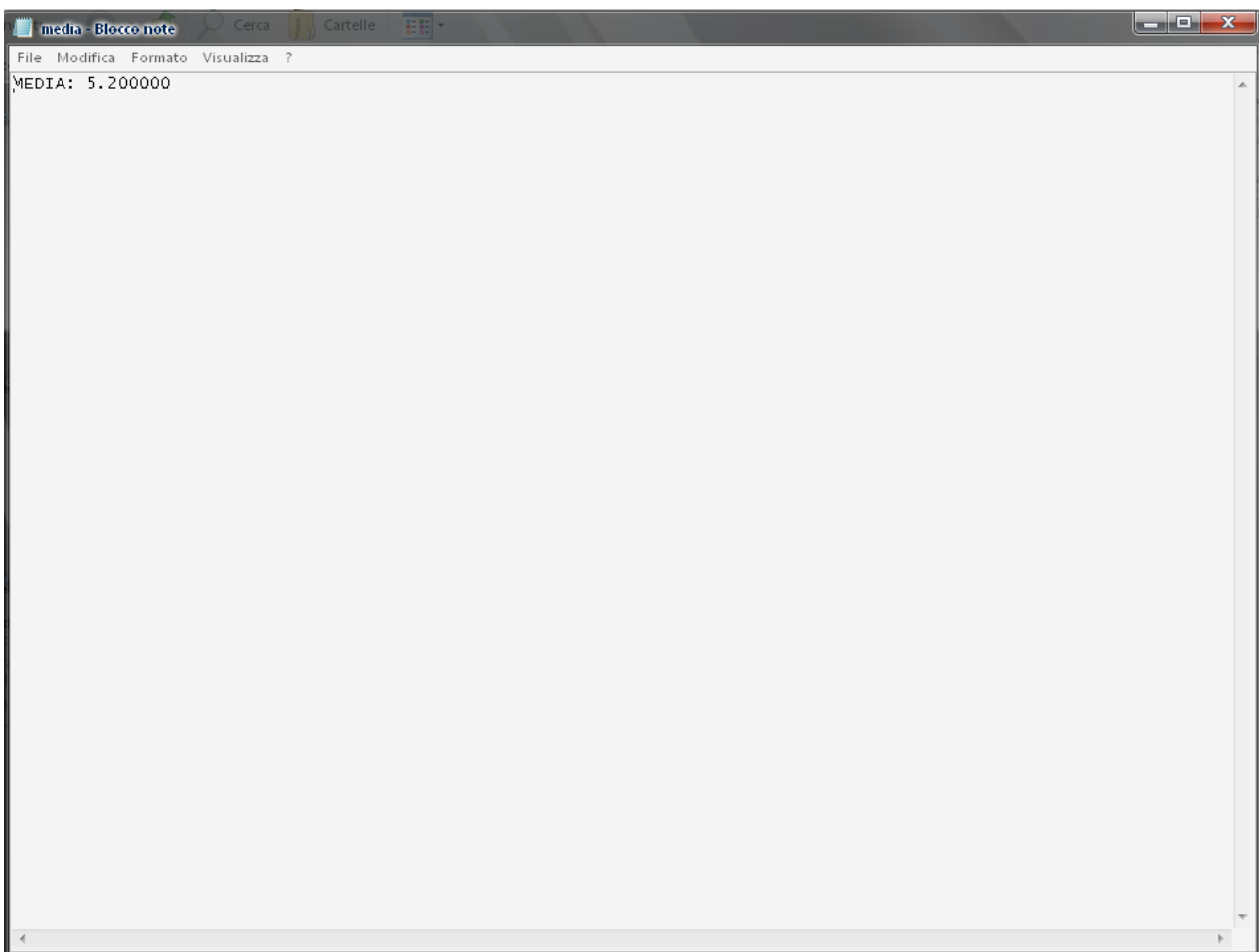
fileOut=fopen(NOME_FILE_OUT,"a");
if(fileOut==NULL)
printf("Impossibile aprire il file di output");
else{
fprintf(fileOut,"MEDIA: %f\n",media);
if(ferror(fileOut)==1)
printf("Errore in scrittura");
if(fclose(fileOut)!=0)
printf("Errore durante la chiusura di un file");}}
if(fclose(fileIn)!=0)
printf("Errore durante la chiusura di un file");
system("pause");
return(0);}}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file5\Debug\file5.exe
3 5 6 7 8 9 2 3 4 5
MEDIA: 5.200000
Premere un tasto per continuare . . .

```



5. Documentazione

6. Testing

FILE N° 6

Un file contiene una sequenza di stringhe formate da ripetizioni dell'unico carattere '*', separate tra loro da uno o più spazi e ritorni a capo. Le stringhe rappresentano una sequenza di interi positivi codificati in codice unario. Ad esempio, il file *****

**** corrisponde alla sequenza 7, 6, 2, 16. Scrivere una funzione C che prende come parametro il nome del file e lo modifica appendendo un fondo al file la media (rappresentata in unario ed arrotondata per difetto) dei valori contenuti nel file. Nell'esempio precedente, all'uscita della funzione il file deve essere il seguente

in cui il valore 7 (cioè la media arrotondata per difetto dei valori 7, 6, 2 e 16) è stato appeso alla fine del file. Si assuma che ogni riga contenga al più 80 caratteri e, conseguentemente, le stringhe siano composte di al massimo 80 caratteri '*'. Si assuma inoltre che i dati nel file siano corretti, ma che il file di ingresso possa non esistere.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Aggiungi Media

Dati d'input: nomefile, nome del file su cui operare

Dati in output: tanti * quanto indica la media degli * presenti in ogni stringa del file

Variabili di lavoro: *fp, i, somma, conta, media, puntatore al file, indice per spazzolare la stringa, somma i caratteri di ogni stringa letta, conta le stringhe lette, contiene somma/conta

main

Dati d'input:

Dati in output:

Costanti: M, N lunghezza massima della stringa contenente *, lunghezza massima nome file

Variabili di lavoro: file, nome del file su cui operare.

3. Costruzione del modello

La funzione che risolve l'esercizio si compone di due fase distinte di lettura e scrittura nel file. Nella prima fase, viene letto il file una stringa alla volta. Sfruttando il fatto che gli spazi e i ritorni a capo vengono ignorati dal formato %s della funzione fscanf(), leggendo il file con tale formato, non è necessario ispezionare il contenuto delle stringhe lette, ma soltanto la loro lunghezza. Per poter utilizzare unicamente le modalità semplici di apertura dei file ("r", "w" e "a"), dopo la lettura il file viene chiuso, per essere successivamente riaperto in modalità di inserimento.

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <string.h>
```

```
#include <stdlib.h>
```

```
#define N 20
```

```
#define M 81
```

```
void AggiungiMedia(char *nomefile);
```

```
int main(void)
```

```
{ char file[N];
```

```
printf("Nome file: ");
```

```
scanf("%s", file);
```

```
AggiungiMedia(file);
```

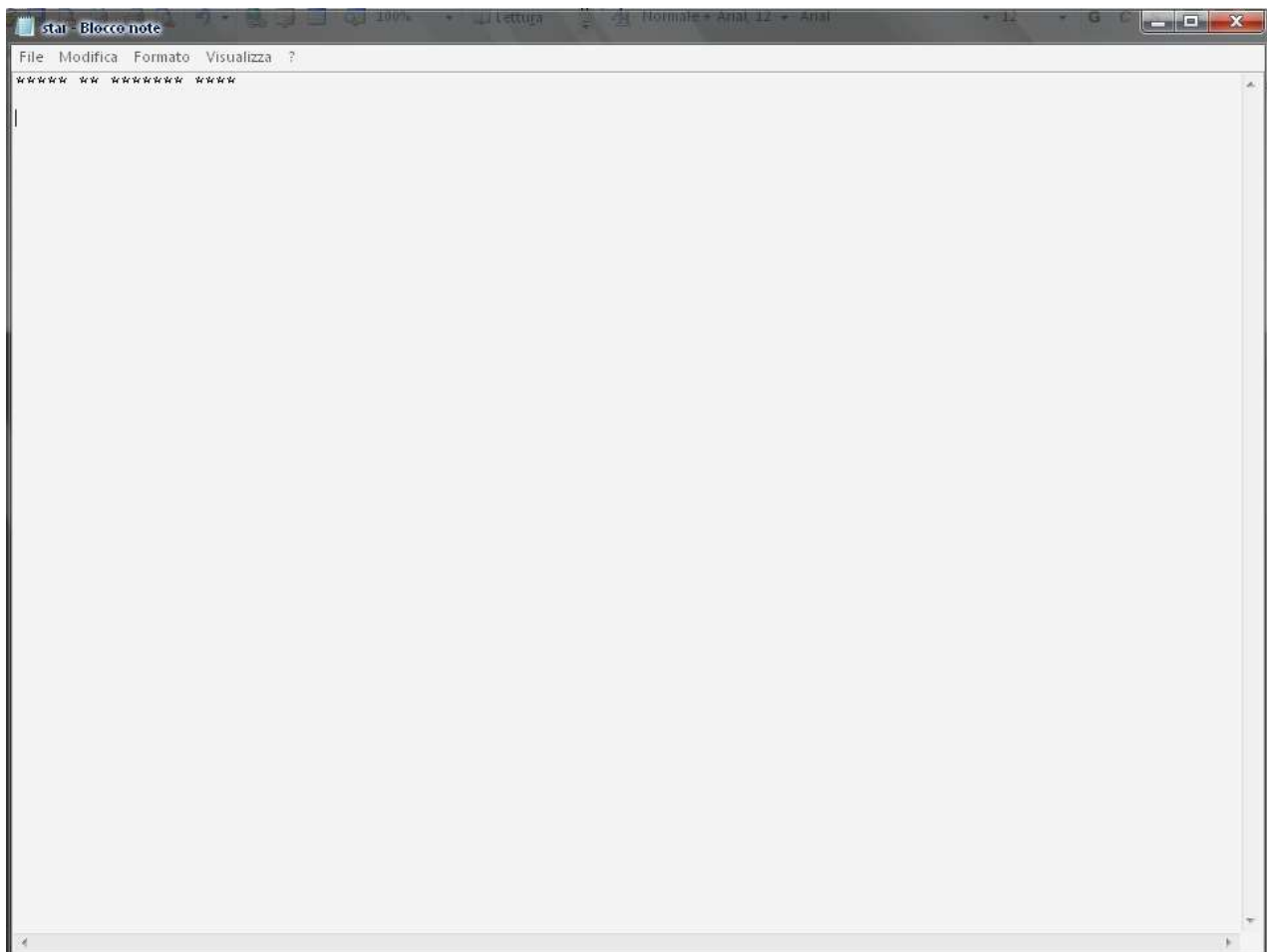
```
system("pause");
```

```

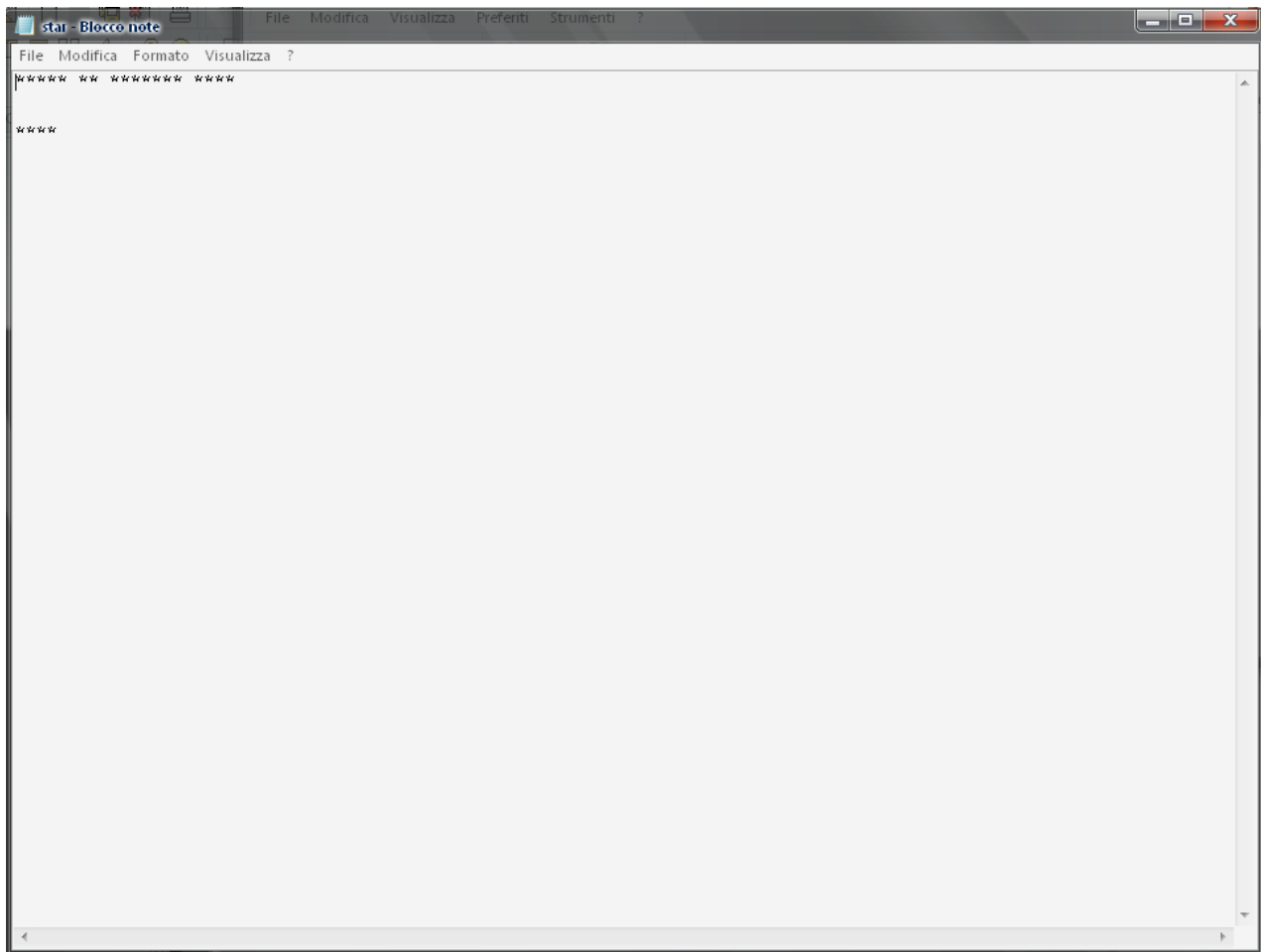
return(0);}
void AggiungiMedia(char *nomefile){
FILE *fp;
int somma = 0, conta = 0, media, i;
char a[M];
if ((fp = fopen(nomefile,"r")) == NULL){
printf("Il file %s non esiste\n",nomefile);
exit(1);}
while (fscanf(fp,"%s",a) == 1){
somma += strlen(a);
conta++;}
media = somma/conta; /* la divisione tra interi avviene con troncamento, restituendo la media arrotondata
per difetto come richiesto */
fclose(fp);
fp = fopen(nomefile,"a");
putc('\n',fp); /* inserisce la media a riga nuova per essere sicuro che entri nelle 80 colonne */
for (i = 0; i < media; i++)
putc('*',fp);
fclose(fp);}

```

Prima dell'esecuzione del programma



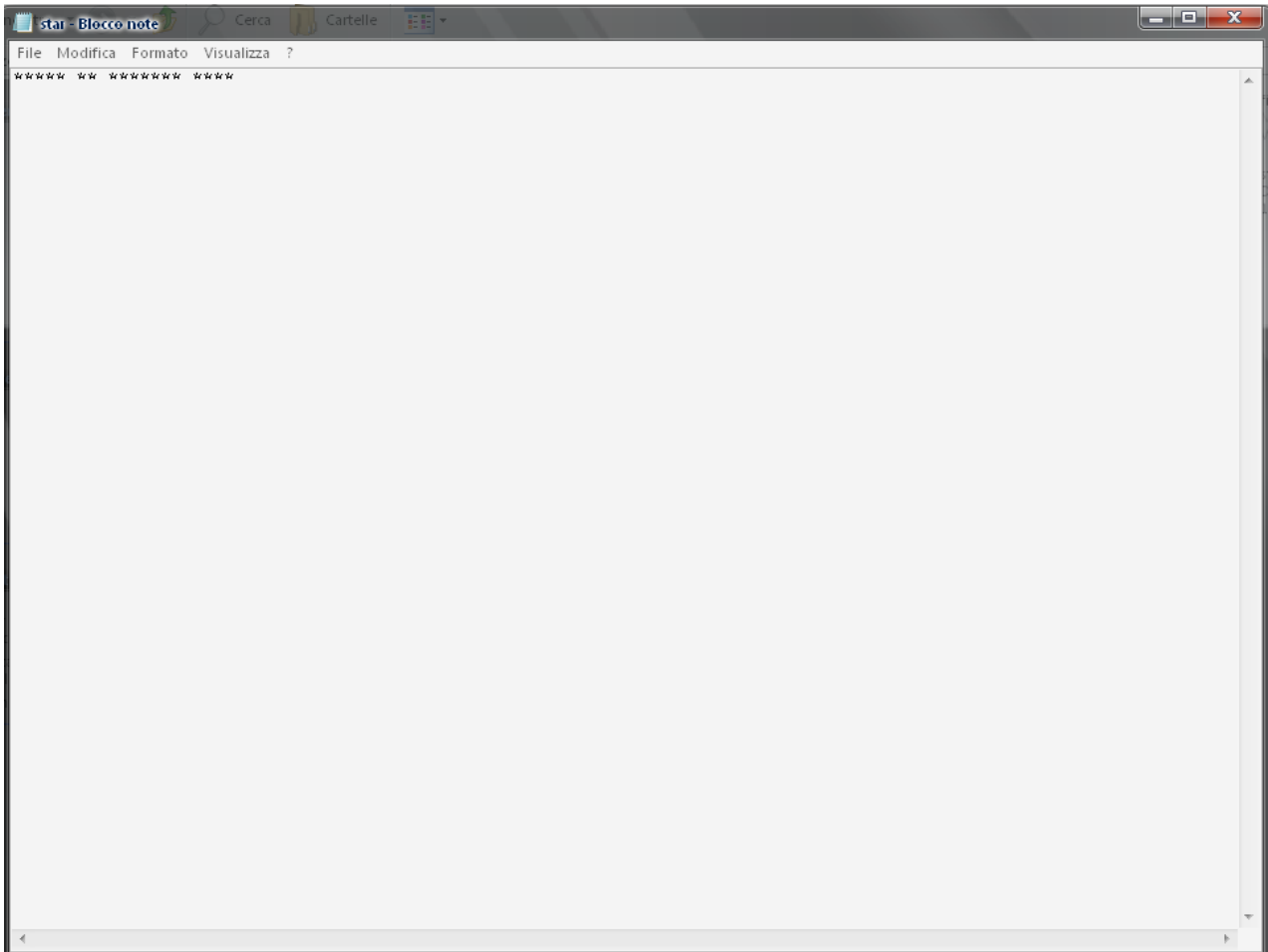
Dopo esecuzione programma il file star.txt è il seguente:

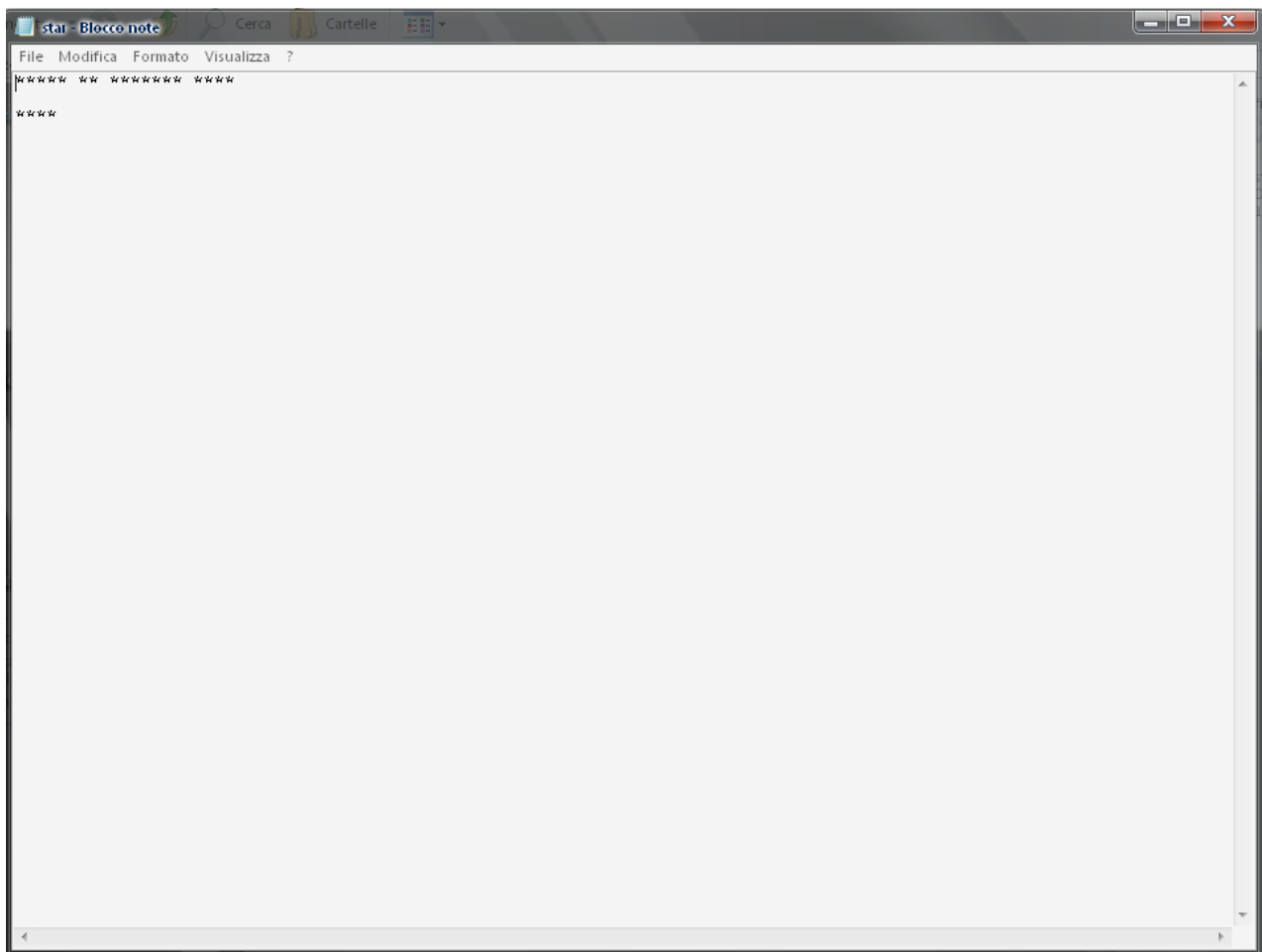


Microsoft Visual C++

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define N 20
#define M 81
void AggiungiMedia(char *nomefile);
int main(void)
{ char file[N];
printf("Nome file: ");
scanf("%s",file);
AggiungiMedia(file);
system("pause");
return(0);}
void AggiungiMedia(char *nomefile){
FILE *fp;
int somma = 0, conta = 0, media, i;
char a[M];
if ((fp = fopen(nomefile,"r")) == NULL){
printf("Il file %s non esiste\n",nomefile);
exit(1);}
while (fscanf(fp,"%s",a) == 1){
somma += strlen(a);
conta++;}
```

```
media = somma/conta; /* la divisione tra interi avviene con troncamento, restituendo la media
arrotondata per difetto come richiesto */
fclose(fp);
fp = fopen(nomefile, "a");
putc('\n', fp); /* inserisce la media a riga nuova per essere sicuro che entri nelle 80 colonne
*/
for (i = 0; i < media; i++)
putc('*', fp);
fclose(fp);}
```





5. Documentazione

6. Testing

FILE N° 7

Si assuma presente in memoria secondaria un file contenente le informazioni relative alle verbalizzazioni di un esame. Il file contiene le coppie nome-voto tra parentesi; il nome è separato dal voto da uno spazio, le coppie tra parentesi sono separate da spazi o ritorni a capo. Ad esempio, il contenuto del file potrebbe essere il seguente:

(gianni 27) (marco 28) (luigi 20) (giovanni 25) (sergio 24) (luisa 29)

Scrivere un programma che utilizzando una funzione, che ricevendo come parametro il nome del file, stampi la media dei voti ottenuti dagli studenti ed il nome e il voto dello studente col voto più alto. Se più studenti condividono il voto più alto si deve stampare il nome del primo studente tra quelli che hanno il voto più alto.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

media

Dati d'input: nomefile, nome del file su cui operare

Dati in output: media dei voti, studente migliore, a parità di voto lo studente inserito per primo
Costanti: N lunghezza massima della stringa contenete il cognome dello studente.

Variabili di lavoro: *p, media, voto, votomax, bravo, studente, app, quanti, per calcolare la media dei voti, per il voto dello studente, per il voto più alto, per il nome dello studente migliore, per il nome dello studente, per inizio dati di un nuovo studente (carattere '(')
)puntatore al file su cui operare, per contare i voti.

3. Costruzione del modello

La scansione del file avviene in un ciclo che legge dal file un carattere alla volta, fino a quando il file non finisce. Se il carattere letto è una parentesi aperta una ulteriore lettura preleva il nome dello studente, il voto corrispondente e consuma la parentesi chiusa senza assegnarla ad una variabile usando il formato di conversione (%*c). Se necessario, vengono aggiornati i valori relativi allo studente più bravo.

4. Codifica

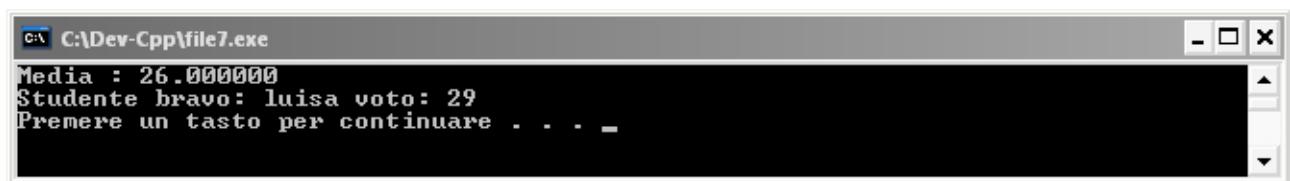
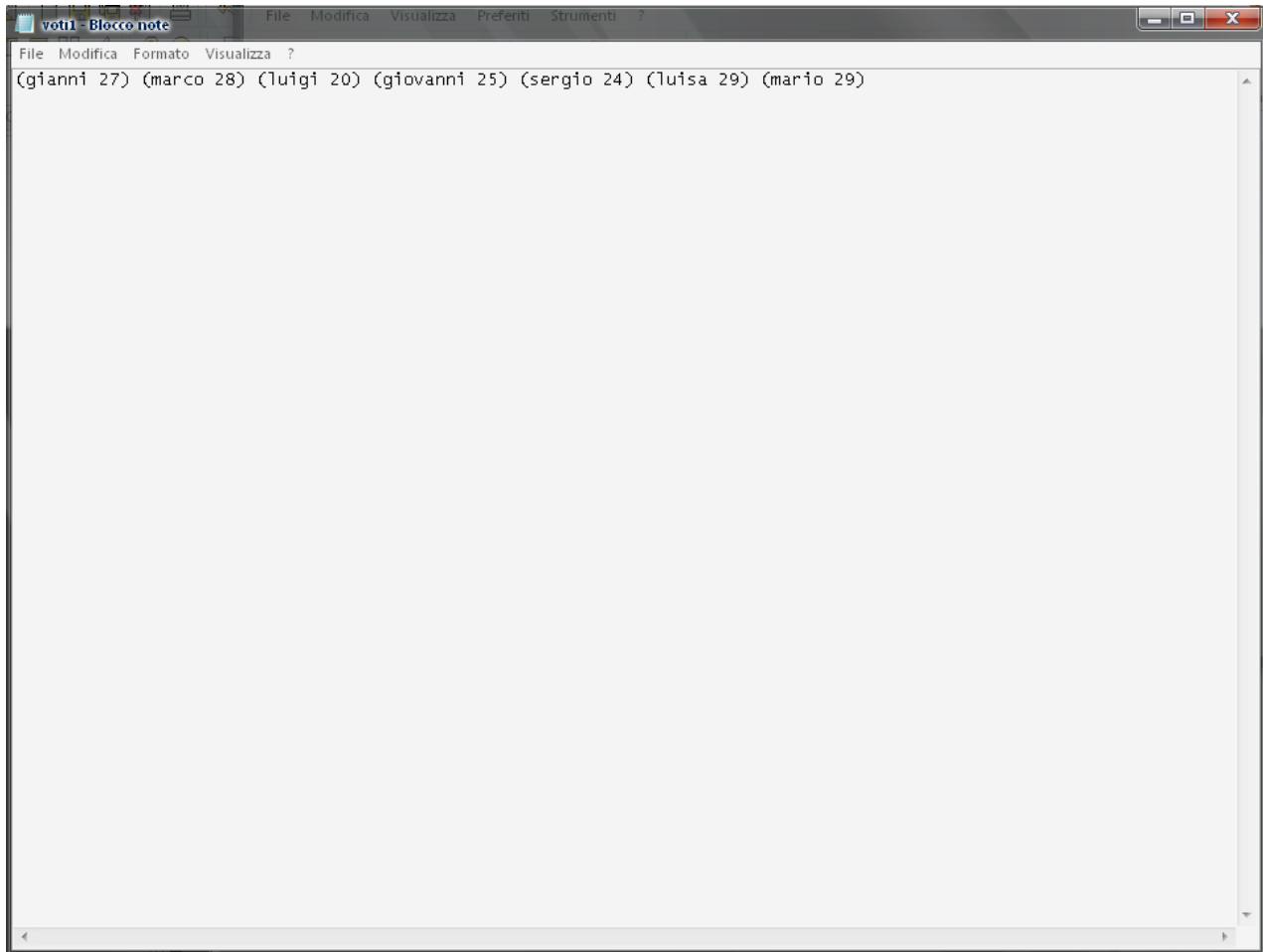
Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include<string.h>
#include<stdlib.h>
#define N 21
void media(char *nomefile);
int main (void) {
    media("voti1.txt");
    system ("pause");
    return(0);}
void media(char *nomefile){
    float media=0.0;
    int quanti, voto, votomax;
    char bravo[N], studente[N ], app;
    FILE *p;
    quanti = votomax = 0;
    p = fopen(nomefile, "r");
    while ( fscanf(p, "%c", &app) != EOF ){
        if (app == '('){
            fscanf(p,"%s%d%*c", studente, &voto);
            media += voto;
```

```

++quanti;
if (voto > votomax){
strcpy(bravo, studente);
votomax = voto;}}}
fclose(p);
printf("Media : %f\n",media/quanti);
printf("Studente bravo: %s voto: %d \n",bravo, votomax);}

```



Microsoft Visual C++

```

#include <stdio.h>
#include<string.h>
#include<stdlib.h>
#define N 21
void media(char *nomefile);
int main (void) {
media("voti1.txt");
system ("pause");
return(0);}
void media(char *nomefile){
float media=0.0;
int quanti, voto, votomax;

```

```

char bravo[N], studente[N ], app;
FILE *p;
quanti = votomax = 0;
p = fopen(nomefile, "r");
while ( fscanf(p, "%c", &app) != EOF ){
if (app == '('){
fscanf(p, "%s%d%*c", studente, &voto);
media += voto;
++quanti;
if (voto > votomax){
strcpy(bravo, studente);
votomax = voto;}}}
fclose(p);
printf("Media : %f\n",media/quanti);
printf("Studente bravo: %s voto: %d \n",bravo, votomax);}

```



```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file7\Debug\file7.exe
Media : 26.000000
Studente bravo: luisa voto: 29
Premere un tasto per continuare . . . _

```

5. Documentazione

6. Testing

FILE N° 8

Un file contiene una sequenza (di lunghezza ignota, possibilmente nulla) di valori reali separati da uno spazio. Come esempio si consideri il seguente file

4.522 5.32 4 5.001 16.2 34.2 45.6

Si scriva una funzione C che riceve in input come parametro il nome del file e sostituisce completamente il contenuto del file stesso con i valori interi ottenuti per arrotondamento (all'intero più vicino) dei valori reali presenti nel file. Ad esempio, dopo l'esecuzione della funzione, il contenuto del file dell'esempio deve essere

5 5 4 5 16 34 46.

Se necessario, si utilizzi la funzione `int Arrotonda(float)` che restituisce l'intero più vicino al valore del parametro di tipo `float`. Si assuma che il file sia presente su disco e che la sequenza sia corretta.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

ArrotondaFile

Dati d'input: `nomefile`, nome del file su cui operare

Dati in output:

Costanti:

Variabili di lavoro: `*fp`, `x`, `*v`, `i`, `n`, puntatore al file su cui operare, valore da arrotondare, puntatore al vettore di interi per memorizzare la sequenza già arrotondata, indice per spazzolare il vettore, numero di elementi del vettore.

Arrotonda

Dati d'input: `x`, valore da arrotondare

Dati in output: intero più vicino al valore del parametro

Costanti:

Variabili di lavoro:

3. Costruzione del modello

La soluzione proposta si basa sull'idea di scorrere il file 3 volte. La prima volta viene calcolato il numero `n` dei reali presenti nel file e viene allocato un vettore di interi della dimensione `n`. Nella seconda scansione i valori letti vengono arrotondati e memorizzati nel vettore. Nella terza scansione i valori nel vettore vengono riscritti nel file (aperto in scrittura). Possibili soluzioni alternative sono:

- ✓ utilizzare un file di appoggio dove copiare i valori letti
- ✓ fare una sola scansione di lettura utilizzando la funzione `realloc()` per il vettore, oppure memorizzando i dati in una lista collegata

4. Codifica

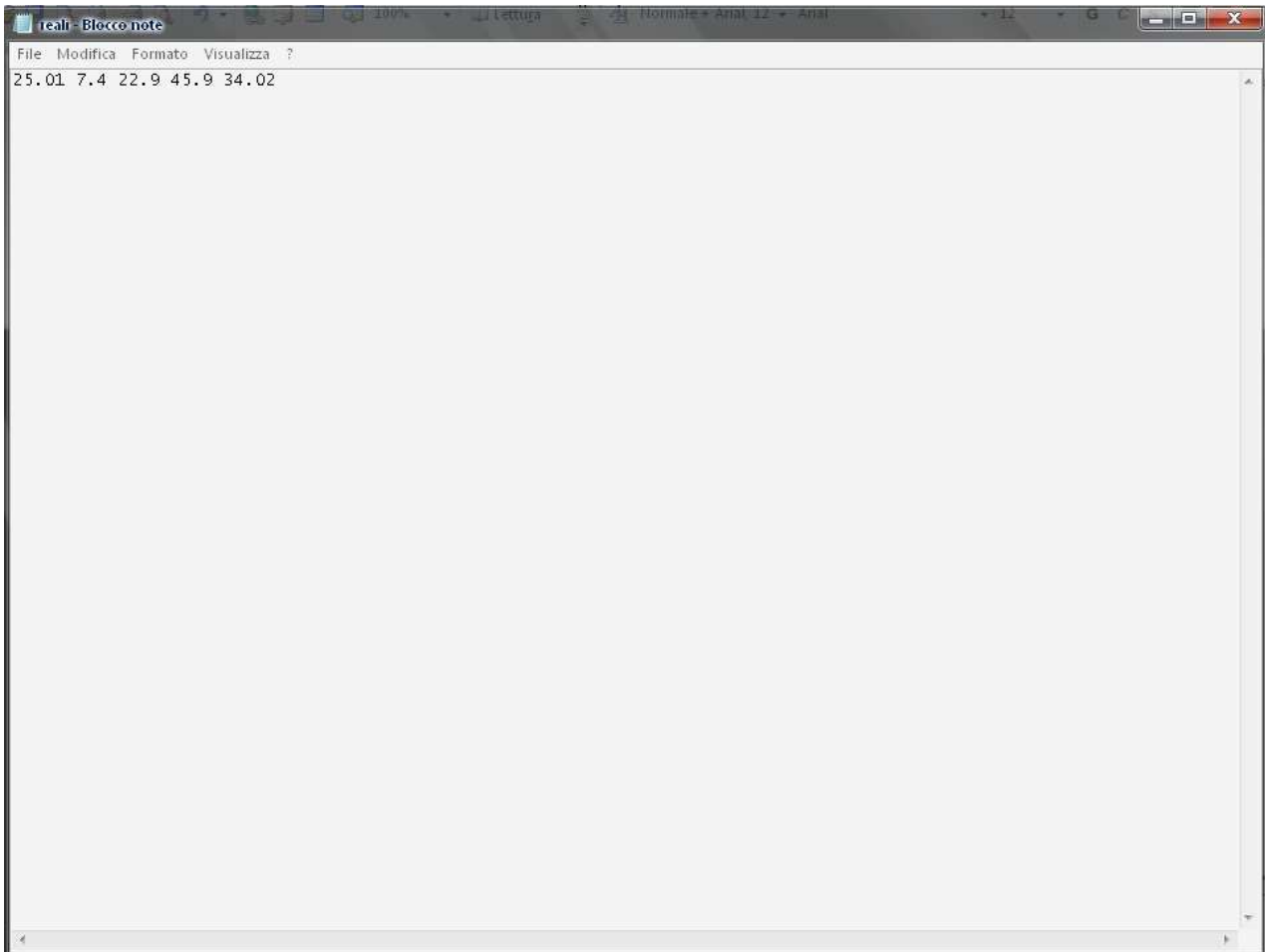
Dev-C++ 4.9.9.2

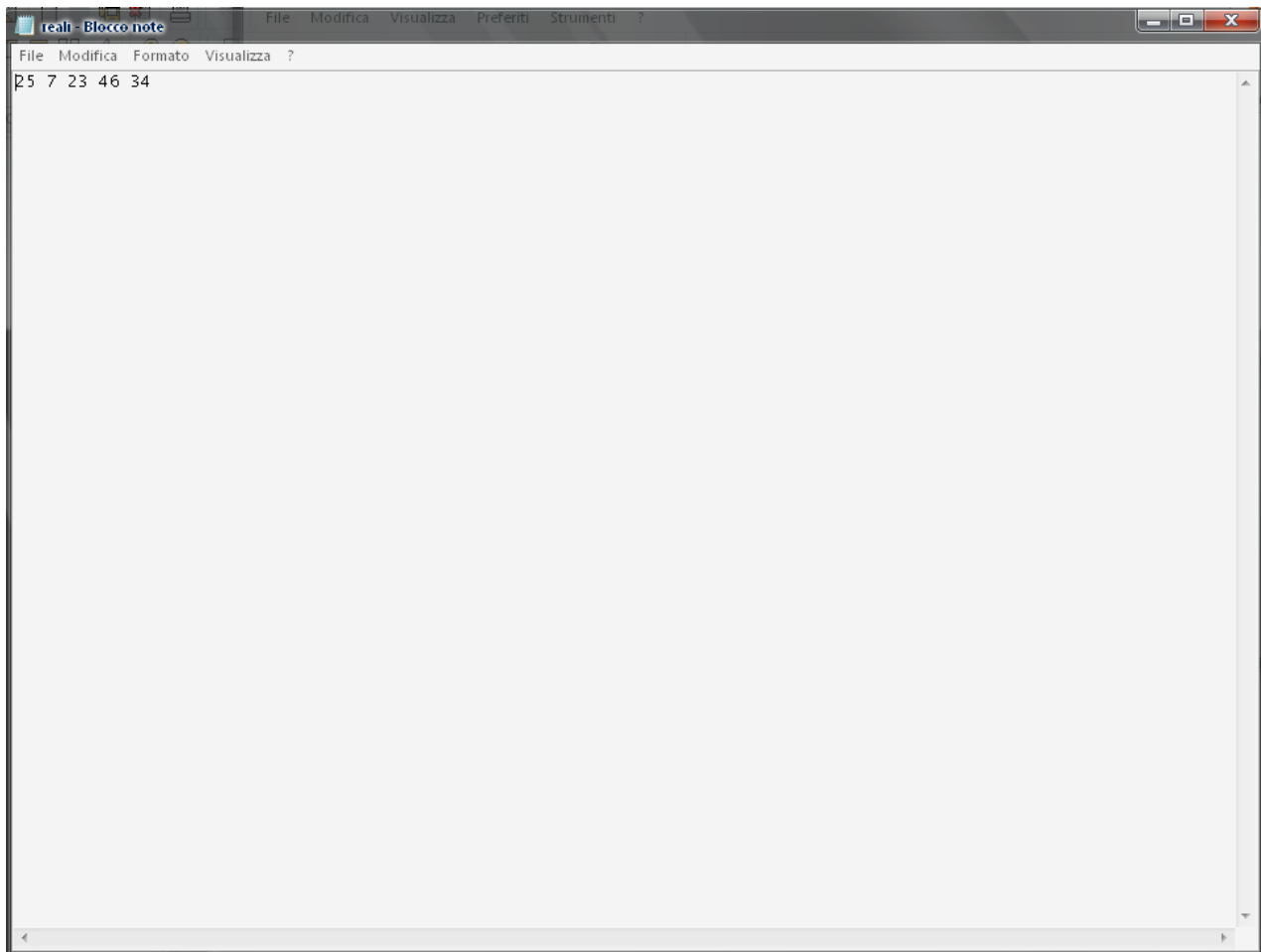
```
#include <stdio.h>
#include<string.h>
#include<stdlib.h>
void ArrotondaFile (char *nomefile);
int Arrotonda (float x);
int main (void) {
    ArrotondaFile("reali.txt");
    printf ("i valori sono stati arrotondati all'intero piu' vicino e scritti sul file reali.txt\n");
    system ("pause");
    return(0);}
int Arrotonda(float x){
```

```

return x + 0.5; } /* conversione automatica da float a int */
void ArrotondaFile (char* nomefile){
float x;
FILE* fp;
int* v; /* vettore di interi per memorizzare la sequenza già arrotondata */
int i = 0, /* indice del vettore */
n = 0; /* numero di elementi nel file */
fp = fopen(nomefile,"r");
while (fscanf(fp,"%f") != EOF) /*lettura dal file del numero di valori float*/
n++;
v = (int*) malloc(n* sizeof(int));
rewind(fp); /*riavvolge il file*/
for (i = 0; i < n; i++){
fscanf(fp,"%f",&x); /*lettura del file*/
v[i] = Arrotonda(x);}
fclose(fp); /*chiusura file*/
fp = fopen(nomefile,"w");
for (i = 0; i < n; i++)
fprintf(fp,"%d ",v[i]); /*scrittura sul file*/
fclose(fp);}

```





Microsoft Visual C++

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
void ArrotondaFile (char *nomefile);
int Arrotonda (float x);
int main (void) {
    ArrotondaFile("reali.txt");
    printf ("i valori sono stati arrotondati all'intero piu' vicino e scritti sul file reali.txt\n");
    system ("pause");
    return(0);}
int Arrotonda(float x){
    return x + 0.5; } /* conversione automatica da float a int */
void ArrotondaFile (char* nomefile){
    float x;
    FILE* fp;
    int* v; /* vettore di interi per memorizzare la sequenza già arrotondata */
    int i = 0, /* indice del vettore */
    n = 0; /* numero di elementi nel file */
    fp = fopen(nomefile, "r");
    while (fscanf(fp, "%f") != EOF) /*lettura dal file del numero di valori float*/
        n++;
    v = (int*) malloc(n * sizeof(int));
    rewind(fp); /*riavvolge il file*/
    for (i = 0; i < n; i++){
        fscanf(fp, "%f", &x); /*letture del file*/
```

```

v[i] = Arrotonda(x);}
fclose(fp); /*chiusura file*/
fp = fopen(nomefile, "w");
for (i = 0; i < n; i++)
fprintf(fp, "%d ", v[i]); /*scrittura sul file*/
fclose(fp);}

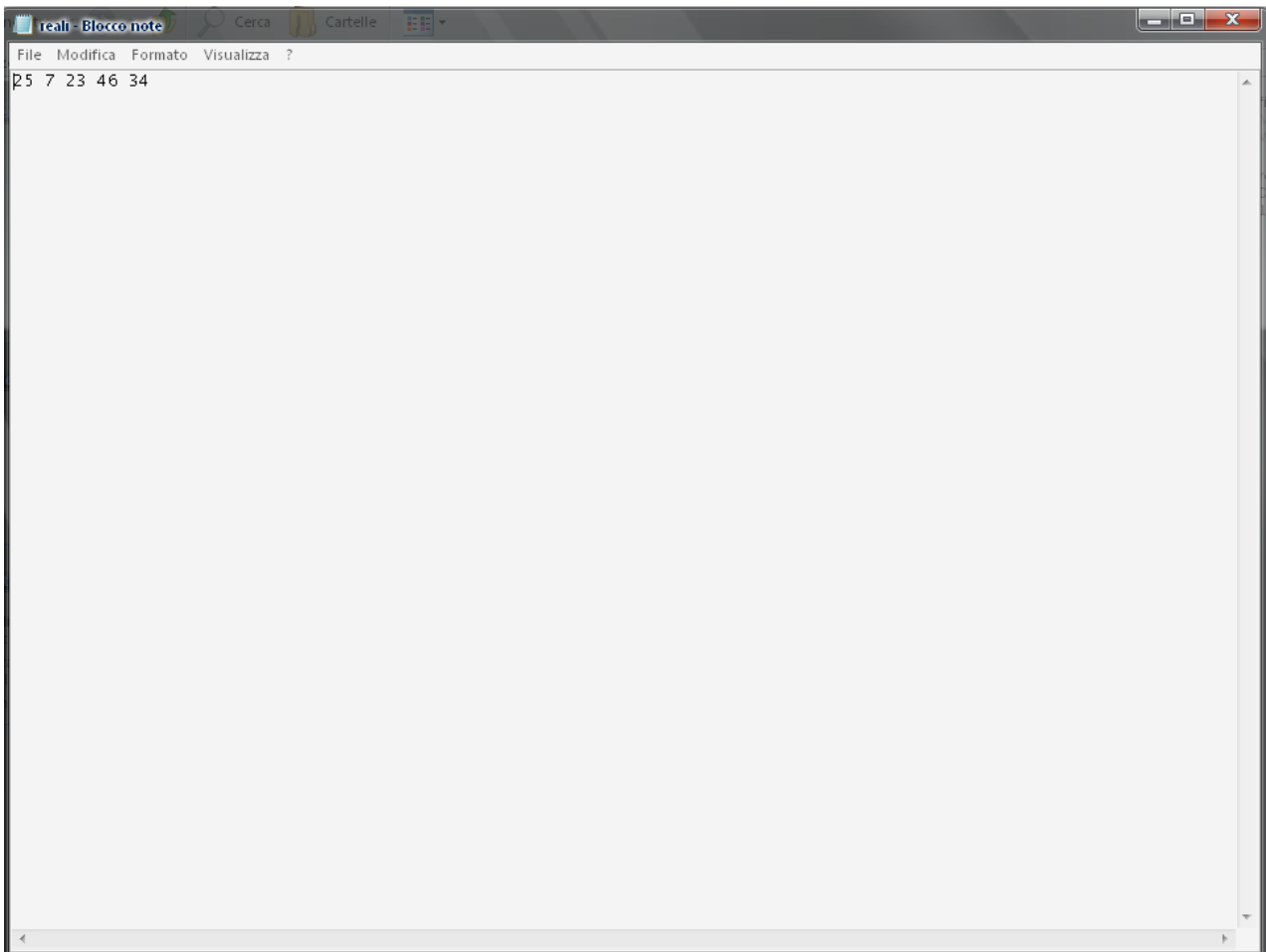
```



```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file8\Debug\file8.exe
i valori sono stati arrotondati all'intero piu' vicino e scritti sul file reali.txt
Premere un tasto per continuare . . .

```



5. Documentazione

6. Testing

FILE N° 9

Cancellare un file il cui nome è ricevuto in input dalla linea di comando.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Dati d'input: *argc*, per identificare il numero di argomenti ricevuti dalla linea di comando, **argv []*, puntatore alle stringhe ricevute dalla linea di comando

Dati in output:

Costanti:

Variabili di lavoro: *risp*, per la scelta dell'utente (s/n)

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include<ctype.h>
#include<conio.h>
#include<stdlib.h>
int main (int argc, char *argv[]) {
char risp;
if (argc!=2)
{printf("sintassi : cancella file\n");
system ("pause");
return (1);}
printf("vuoi cancellare il file %s (s/n)", argv[1]);
risp=getche();
if(tolower(risp)=='s')
if(remove(argv[1])){
printf("impossibile cancellare il file");
system ("pause");
return (2);}
system ("pause");
return(0);}
```



Microsoft Visual C++

```
#include <stdio.h>
#include<ctype.h>
#include<conio.h>
#include<stdlib.h>
int main (int argc, char *argv[]) {
char risp;
if (argc!=2)
{printf("sintassi : cancella file\n");
system ("pause");
```

```
return (1);}
printf("vuoi cancellare il file %s(s/n)", argv[1]);
resp=getche();
if(tolower(resp)=='s')
if(remove(argv[1])){
printf("impossibile cancellare il file");
system ("pause");
return (2);}
system ("pause");
return(0);}
```



5. Documentazione

6. Testing

FILE N° 1 (FILE DATI)

Si scriva un sottoprogramma che modifica i dati relativi ad un rilievo altimetrico, caratterizzato da ascissa e ordinata (struct punto) e da altezza, memorizzato su file. Il sottoprogramma individua il rilievo da modificare e interagisce con l'utente per ottenere i nuovi dati da inserire nel rilievo.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

inserisciNuovoRilievo

Dati d'input: ril ,arch, di tipo rilievo (struct contenente altezza e coordinate (ascissa e ordinata, contenute a loro volta nella struct punto)), nome del file su cui operare

Dati in output:

Variabili di lavoro: *f, puntatore al file

modificaRilievo

Dati d'input: p ,arch, di tipo punto (struct punto per determinare ascissa e ordinata), nome del file su cui operare

Dati in output:

Variabili di lavoro: *f, ril, trovato, puntatore al file, di tipo rilievo (struct contenente altezza e coordinate (ascissa e ordinata, contenute a loro volta nella struct punto)), di tipo booleano per determinare il rilievo da modificare

stampaRilievi

Dati d'input: arch, nome del file su cui operare

Dati in output: rilievi inseriti nel file

Variabili di lavoro: *f, ril, puntatore al file, di tipo rilievo (struct contenente altezza e coordinate (ascissa e ordinata, contenute a loro volta nella struct punto))

main

Dati d'input:

Dati in output:

Variabili di lavoro: r, scelta, nfile, di tipo rilievo (struct contenente altezza e coordinate (ascissa e ordinata, contenute a loro volta nella struct punto)), scelta per determinare l'operazione da effettuare, nome del file su cui operare.

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int x;
    int y;} punto;
typedef struct {
    int altezza;
    punto coord;} rilievo;
typedef char string[20];
typedef enum {FALSO, VERO} boolean;
void inserisciNuovoRilievo(rilievo r, string arch);
void modificaRilievo(punto p, string arch);
void stampaRilievi(string arch);
int main(void)
{string nfile = "arch.txt";
```

```

rilievo r;
int scelta;
scelta=0;
while(scelta!=3){
printf("*****\n");
printf("***** MENU PRINCIPALE *****\n");
printf("*****\n\n");
printf("\t1) inserisci\n");
printf("\t2) stampa\n");
printf("\t3) ricerca\n");
printf("\t4) modifica\n");
printf("\t5) esci\n\n");
printf("Cosa vuoi fare? ");
scanf("%d",&scelta);
switch(scelta){
case 1:printf("coordinate punto: \n");
scanf("%d", &r.coord.x);
scanf("%d", &r.coord.y);
printf("Altezza: \n");
scanf("%d", &r.altezza);
inserisciNuovoRilievo(r, nfile);
break;
case 2:stampaRilievi(nfile);
break;
case 3:printf("Non implem.\n");
break;
case 4:printf("coordinate punto: \n");
scanf("%d", &r.coord.x);
scanf("%d", &r.coord.y);
modificaRilievo(r.coord, nfile);
break;
case 5:exit(0);
break;
default:printf("digita un'altra lettera\n");}}
system("pause");
return(0);}

void inserisciNuovoRilievo(rilievo ril, string arch){
FILE *f;
f=fopen(arch,"ab");
if(f==NULL)
printf("Impossibile aprire il file");
else{
if (fwrite(&ril, sizeof(rilievo), 1, f)!=1)
printf("errore nella scrittura su file\n");
fclose(f);}}

void stampaRilievi(string arch){
rilievo ril;
FILE *f;
f=fopen(arch,"rb");
if(f==NULL){
printf("Impossibile aprire il file");}
else{
while(fread(&ril, sizeof(rilievo), 1, f)!=0)
printf("Altezza alle coordinate (%d, %d): %d\n",

```

```

ril.coord.x, ril.coord.y, ril.altezza);
fclose(f);}}
void modificaRilievo(punto p, string arch){
rilievo ril;
FILE *f;
boolean trovato;
f=fopen(arch,"rb+");
if(f==NULL){
printf("Impossibile aprire il file");}
else{
trovato = FALSO;
while(!trovato && fread(&ril, sizeof(rilievo), 1, f)!=0){
if(ril.coord.x == p.x && ril.coord.y == p.y)
trovato = VERO;}
if(trovato){
printf("Altezza = %d\n", ril.altezza);
printf("Inserisci una nuova altezza\n");
scanf("%d", &ril.altezza);
fseek(f, -sizeof(rilievo), SEEK_CUR);
if (fwrite(&ril, sizeof(rilievo), 1, f)!=1)
printf("errore nella scrittura su file\n");}
else printf("Elemento non trovato\n");
fclose(f);}}

```

```

C:\Dev-Cpp\file1dat.exe
*****
***** MENU PRINCIPALE *****
*****

1) inserisci
2) stampa
3) ricerca
4) modifica
5) esci

Cosa vuoi fare? 1
coordinate punto:
23
24
Altezza:
23
*****
***** MENU PRINCIPALE *****
*****

1) inserisci
2) stampa
3) ricerca
4) modifica
5) esci

Cosa vuoi fare? 2
Altezza alle coordinate <12, 12>: 23
Altezza alle coordinate <23, 24>: 23
Altezza alle coordinate <23, 24>: 23
*****
***** MENU PRINCIPALE *****
*****

1) inserisci
2) stampa
3) ricerca
4) modifica
5) esci

Cosa vuoi fare?

```

Microsoft Visual C++

```

#include <stdio.h>
#include<stdlib.h>
typedef struct {
int x;
int y;} punto;
typedef struct {
int altezza;
punto coord;} rilievo;
typedef char string[20];
typedef enum {FALSO, VERO} boolean;
void inserisciNuovoRilievo(rilievo r, string arch);
void modificaRilievo(punto p, string arch);
void stampaRilievi(string arch);
int main(void)
{string nfile = "arch.txt";
rilievo r;
int scelta;
scelta=0;
while(scelta!=3){
printf("*****\n");
printf("***** MENU PRINCIPALE *****\n");
printf("*****\n\n");
printf("\t1) inserisci\n");
printf("\t2) stampa\n");
printf("\t3) ricerca\n");
printf("\t4) modifica\n");
printf("\t5) esci\n\n");
printf("Cosa vuoi fare? ");
scanf("%d",&scelta);
switch(scelta){
case 1:printf("coordinate punto: \n");
scanf("%d", &r.coord.x);
scanf("%d", &r.coord.y);
printf("Altezza: \n");
scanf("%d", &r.altezza);
inserisciNuovoRilievo(r, nfile);
break;
case 2:stampaRilievi(nfile);
break;
case 3:printf("Non implem.\n");
break;
case 4:printf("coordinate punto: \n");
scanf("%d", &r.coord.x);
scanf("%d", &r.coord.y);
modificaRilievo(r.coord, nfile);
break;
case 5:exit(0);
break;
default:printf("digita un'altra lettera\n");}}
system("pause");
return(0);}
void inserisciNuovoRilievo(rilievo ril, string arch){
FILE *f;
f=fopen(arch,"ab");

```

```

if(f==NULL)
printf("Impossibile aprire il file");
else{
if (fwrite(&ril, sizeof(rilievo), 1, f)!=1)
printf("errore nella scrittura su file\n");
fclose(f);}}
void stampaRilievi(string arch){
rilievo ril;
FILE *f;
f=fopen(arch, "rb");
if(f==NULL){
printf("Impossibile aprire il file");}
else{
while(fread(&ril, sizeof(rilievo), 1, f)!=0)
printf("Altezza alle coordinate (%d, %d): %d\n",
ril.coord.x, ril.coord.y, ril.altezza);
fclose(f);}}
void modificaRilievo(punto p, string arch){
rilievo ril;
FILE *f;
boolean trovato;
f=fopen(arch, "rb+");
if(f==NULL){
printf("Impossibile aprire il file");}
else{
trovato = FALSO;
while(!trovato && fread(&ril, sizeof(rilievo), 1, f)!=0){
if(ril.coord.x == p.x && ril.coord.y == p.y)
trovato = VERO;}
if(trovato){
printf("Altezza = %d\n", ril.altezza);
printf("Inserisci una nuova altezza\n");
scanf("%d", &ril.altezza);
fseek(f, -sizeof(rilievo), SEEK_CUR);
if (fwrite(&ril, sizeof(rilievo), 1, f)!=1)
printf("errore nella scrittura su file\n");}
else printf("Elemento non trovato\n");
fclose(f);}}

```

```
c:\documents and settings\hp\documenti\visual studio 2010\Projects\file1bin\Debug\file1bin.exe
***** MENU PRINCIPALE *****
*****
1) inserisci
2) stampa
3) ricerca
4) modifica
5) esci

Cosa vuoi fare? 2
Altezza alle coordinate (3, 4): 16
Altezza alle coordinate (4, 5): 14
***** MENU PRINCIPALE *****
*****
1) inserisci
2) stampa
3) ricerca
4) modifica
5) esci

Cosa vuoi fare? 3
Non implem.
Premere un tasto per continuare . . .
```

5. Documentazione

6. Testing

FILE N° 2

Si sviluppi un programma che consenta ad una azienda di mantenere un archivio dei propri impiegati. Il programma verrà utilizzato dal responsabile della gestione del personale dell'azienda oppure da un suo delegato e deve consentire:

- ✓ L'immissione dei dati di un nuovo impiegato.
- ✓ La cancellazione dei dati relativi ad un impiegato che non lavora più per l'azienda in questione
- ✓ La ricerca dei dati di un impiegato a partire dalla conoscenza del numero di matricola
- ✓ L'aggiornamento dello stipendio di ciascun impiegato. Si suppone che l'aggiornamento venga effettuato
- ✓ incrementando della stessa quantità percentuale lo stipendio degli impiegati.
- ✓ La stampa degli stipendi degli impiegati per la gestione delle buste paga.
- ✓ La stampa di tutte le informazioni in archivio.

Le informazioni relative agli impiegati e rilevanti per l'azienda sono il nome e l'indirizzo dell'impiegato, lo stipendio ed il numero di matricola. Lo stipendio viene assegnato all'impiegato al momento della sua assunzione e viene incrementato periodicamente (insieme a quello di tutti gli altri impiegati) utilizzando la funzionalità di aggiornamento degli stipendi. Il programma deve garantire che il numero di matricola sia univoco per ogni impiegato. Infine, il programma deve conservare in modo persistente le informazioni relative agli impiegati.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

inserisci

Dati d'input: **a di tipo archivioImp, i di tipo impiegato*

Dati in output: falso se archivio è pieno, falso altrimenti

Costanti: *MAX_NUM_CHAR, MAX_ARCH, ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro:

aggiornastipendi

Dati d'input: **a di tipo archivioImp, incr di tipo float*

Dati in output:

Costanti: *MAX_NUM_CHAR, MAX_ARCH, ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro: *i*, per spazzolare l'archivio degli impiegati

stampaarchivio

Dati d'input: *a di tipo archivioImp*

Dati in output:

Costanti: *MAX_NUM_CHAR, MAX_ARCH, ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro: *i* indice per spazzolare l'archivio

leggiImpiegato

Dati d'input:

Dati in output: *x di tipo impiegato*

Costanti: *MAX_NUM_CHAR, MAX_ARCH, ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro:

stampaMenu

Dati d'input:

Dati in output: stampa le possibili operazioni da effettuare sull'archivio

Costanti: *MAX_NUM_CHAR, MAX_ARCH, ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro: :

caricaArchivio

Dati d'input: *a di tipo *archivioImp*

Dati in output:

Costanti: *MAX_NUM_CHAR*, *MAX_ARCH*, *ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro: *f, puntatore al file su cui operare, numE lem di tipo int

salva Archivio

Dati d'input: a di tipo *archivioImp*

Dati in output: a di tipo *archivioImp* viene scritta sul file

Costanti: *MAX_NUM_CHAR*, *MAX_ARCH*, *ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro: *f, puntatore al file su cui operare

main:

Dati d'input:

Dati in output:

Costanti: *MAX_NUM_CHAR*, *MAX_ARCH*, *ARCH_NAME*, dimensione massima nome, indirizzo, dimensione massima array di impiegati, nome del file su cui operare

Variabili di lavoro: *archivio* di tipo **archivioImp**, struct per memorizzare l'array di tipo **impiegato** (struct contenente nome, indirizzo, matricola e stipendio di ogni impiegato) e il numero dell'impiegato *numImp*, *imp* struct di tipo **impiegato**, *risultato* di tipo booleano per verificare se l'archivio è pieno, scelta per determinare l'operazione da effettuare.

3. Costruzione del modello

Non sono state implementate le funzioni stampastipendio e cancella, che sono lasciate allo studente per esercizio

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
#include<stdlib.h>
#define MAX_NUM_CHAR 20
#define MAX_ARCH 100
#define ARCH_NAME "archImp.arc"
/* definizione dei tipi */
typedef char stringa[MAX_NUM_CHAR];
typedef struct{
    stringa nome;
    stringa indirizzo;
    int matricola;
    float stipendio;} impiegato;
typedef struct{
    impiegato arch[MAX_ARCH];
    int numImp;} archivioImp;
typedef enum {falso, vero} boolean;
boolean inserisci(archivioImp *a, impiegato i);
boolean cancella(archivioImp *a, int m);
void aggiornaStipendi(archivioImp *a, float incr);
void stampaStipendi(archivioImp a);
void stampaArchivio(archivioImp a);
void stampaMenu();
impiegato leggiImpiegato();
void caricaArchivio(archivioImp *a);
void salvaArchivio(archivioImp a);
```

```

int main(void)
{int scelta;
archivioImp archivio;
impiegato imp;
boolean risultato;
float stip;
/* a->numImp = 0; */
caricaArchivio(&archivio);
/* stampa menu */
stampaMenu();
/* acquisizione scelta utente */
scanf("%d", &scelta);
/* interpretazione scelta ed esecuzione della relativa operazione */
while(scelta!=6){
if(scelta == 1){
imp = leggiImpiegato();
risultato = inserisci(&archivio, imp);
if(risultato == falso)
printf("Errore: archivio pieno\n");
else if(scelta == 2)
printf("la scelta e` cancella (funzione da implementare\n");
else if (scelta == 3){
printf("Inserisci l'incremento percentuale di stipendio: ");
scanf("%f", &stip);
aggiornaStipendi(&archivio, stip);}
else if(scelta == 4)
printf("la scelta e` stampa stipendi (funzione da implementare)\n");
else if(scelta == 5)
stampaArchivio(archivio);
stampaMenu();
scanf("%d", &scelta);}
salvaArchivio(archivio);
system("pause");
return(0);}
void stampaMenu(){
printf("Scegli una delle seguenti opzioni: \n");
printf("1. Inserisci\n");
printf("2. Cancella\n");
printf("3. Aggiorna stipendi\n");
printf("4. Stampa stipendi\n");
printf("5. Stampa contenuto archivio\n");
printf("6. Termina il programma\n");}
impiegato leggiImpiegato(){
impiegato x;
printf("Inserisci il nome: ");
scanf("%s", x.nome);
printf("Inserisci il indirizzo: ");
scanf("%s", x.indirizzo);
printf("Inserisci lo stipendio: ");
scanf("%f", &x.stipendio);
return (x);}
boolean inserisci(archivioImp *a, impiegato i){
if(a->numImp == MAX_ARCH)
return falso;

```

```

else{
i.matricola = a->numImp+1;
/* (*a).arch[(*a).numImp] = i; */
a->arch[a->numImp] = i;
/* (*a).numImp = (*a).numImp + 1; */
a->numImp = a->numImp + 1;
return vero;}}
boolean cancella(archivioImp *a, int m){
void aggiornaStipendi(archivioImp *a, float incr){
int i;
for(i=0; i < a->numImp; i++){
a->arch[i].stipendio = a->arch[i].stipendio + (a->arch[i].stipendio * incr);}
void stampaStipendi(archivioImp a){/*da implementare*/
}
void stampaArchivio(archivioImp a){
int i;
for(i=0;i<a.numImp;i++){
printf("%s, %s, %f, %d\n",
a.arch[i].nome, a.arch[i].indirizzo,
a.arch[i].stipendio, a.arch[i].matricola);}}
void caricaArchivio(archivioImp *a){
FILE *f; int numElem;
f = fopen(ARCH_NAME, "rb");
if(f == NULL){
printf("Archivio non esistente o corrotto, ne creo uno nuovo\n");
a->numImp = 0;}
else{
if(fread(&numElem, sizeof(int), 1, f)<1){
printf("Archivio corrotto, ne creo uno nuovo\n");
a->numImp = 0;}
else{
a->numImp = numElem;
if(fread(a->arch, sizeof(impiegato), numElem, f)<numElem){
printf("Archivio corrotto, ne creo uno nuovo\n");
a->numImp = 0;}}
fclose(f);}}
void salvaArchivio(archivioImp a){
FILE *f;
f = fopen(ARCH_NAME, "wb");
if(f==NULL)
printf("Impossibile salvare i dati in archivio. I dati verranno persi\n");
else{
if(fwrite(&a.numImp, sizeof(int), 1, f)!=1)
printf("Errore di scrittura nel file. I dati potranno essere persi\n");
else if(fwrite(a.arch, sizeof(impiegato), a.numImp, f)!=a.numImp)
printf("Errore di scrittura nel file. I dati potranno essere persi\n");
fclose(f);}}

```

```
C:\Dev-Cpp\file2dat.exe
Archivio non esistente o corrotto, ne creo uno nuovo
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
1
Inserisci il nome: mario
Inserisci il indirizzo: bari
Inserisci lo stipendio: 1400
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
2
la scelta e' cancella <funzione da implementare
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
3
Inserisci l'incremento percentuale di stipendio: 20
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
5
mario, bari, 29400.000000, 1
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
6
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

```
#include <stdio.h>
#include<stdlib.h>
#define MAX_NUM_CHAR 20
#define MAX_ARCH 100
#define ARCH_NAME "archImp.arc"
/* definizione dei tipi */
typedef char stringa[MAX_NUM_CHAR];
typedef struct{
    stringa nome;
    stringa indirizzo;
    int matricola;
    float stipendio;} impiegato;
typedef struct{
    impiegato arch[MAX_ARCH];
    int numImp;} archiviolmp;
typedef enum {falso, vero} boolean;
boolean inserisci(archiviolmp *a, impiegato i);
void cancella(archiviolmp *a, int m);
void aggiornaStipendi(archiviolmp *a, float incr);
void stampaStipendi(archiviolmp a);
void stampaArchivio(archiviolmp a);
void stampaMenu();
impiegato leggiImpiegato();
void caricaArchivio(archiviolmp *a);
void salvaArchivio(archiviolmp a);
int main(void)
{int scelta;
archiviolmp archivio;
impiegato imp;
boolean risultato;
float stip;
/* a->numImp = 0; */
caricaArchivio(&archivio);
/* stampa menu */
stampaMenu();
/* acquisizione scelta utente */
scanf("%d", &scelta);
/* interpretazione scelta ed esecuzione della relativa operazione */
while(scelta!=6){
if(scelta == 1){
imp = leggiImpiegato();
risultato = inserisci(&archivio, imp);
if(risultato == falso)
printf("Errore: archivio pieno\n");
else if(scelta == 2)
printf("la scelta e` cancella (funzione da implementare\n");
else if (scelta == 3){
printf("Inserisci l'incremento percentuale di stipendio: ");
scanf("%f", &stip);
aggiornaStipendi(&archivio, stip);}
else if(scelta == 4)
printf("la scelta e` stampa stipendi (funzione da implementare)\n");
else if(scelta == 5)
```

```

stampaArchivio(archivio);
stampaMenu();
scanf("%d", &scelta);}
salvaArchivio(archivio);
system("pause");
return(0);}
void stampaMenu(){
printf("Scegli una delle seguenti opzioni: \n");
printf("1. Inserisci\n");
printf("2. Cancella\n");
printf("3. Aggiorna stipendi\n");
printf("4. Stampa stipendi\n");
printf("5. Stampa contenuto archivio\n");
printf("6. Termina il programma\n");}
impiegato leggiImpiegato(){
impiegato x;
printf("Inserisci il nome: ");
scanf("%s", x.nome);
printf("Inserisci il indirizzo: ");
scanf("%s", x.indirizzo);
printf("Inserisci lo stipendio: ");
scanf("%f", &x.stipendio);
return (x);}
boolean inserisci(archivioImp *a, impiegato i){
if(a->numImp == MAX_ARCH)
return falso;
else{
i.matricola = a->numImp+1;
/* (*a).arch[(*a).numImp] = i; */
a->arch[a->numImp] = i;
/* (*a).numImp = (*a).numImp + 1; */
a->numImp = a->numImp + 1;
return vero;}}
void cancella(archivioImp *a, int m){}
void aggiornaStipendi(archivioImp *a, float incr){
int i;
for(i=0; i < a->numImp; i++)
a->arch[i].stipendio = a->arch[i].stipendio + (a->arch[i].stipendio * incr);}
void stampaStipendi(archivioImp a){
}
void stampaArchivio(archivioImp a){
int i;
for(i=0; i<a.numImp; i++){
printf("%s, %s, %f, %d\n",
a.arch[i].nome, a.arch[i].indirizzo,
a.arch[i].stipendio, a.arch[i].matricola);}}
void caricaArchivio(archivioImp *a){
FILE *f; int numElem;
f = fopen(ARCH_NAME, "rb");
if(f == NULL){
printf("Archivio non esistente o corrotto, ne creo uno nuovo\n");
a->numImp = 0;}
else{
if(fread(&numElem, sizeof(int), 1, f)<1){

```

```

printf("Archivio corrotto, ne creo uno nuovo\n");
a->numImp = 0;}
else{
a->numImp = numElem;
if(fread(a->arch, sizeof(impiegato), numElem, f)<numElem){
printf("Archivio corrotto, ne creo uno nuovo\n");
a->numImp = 0;}}
fclose(f);}}
void salvaArchivio(archivioImp a){
FILE *f;
f = fopen(ARCH_NAME, "wb");
if(f==NULL)
printf("Impossibile salvare i dati in archivio. I dati verranno persi\n");
else{
if(fwrite(&a.numImp, sizeof(int), 1, f)!=1)
printf("Errore di scrittura nel file. I dati potranno essere persi\n");
else if(fwrite(a.arch, sizeof(impiegato), a.numImp, f)!=a.numImp)
printf("Errore di scrittura nel file. I dati potranno essere persi\n");
fclose(f);}}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file2bin\Debug\file2bin.exe
Archivio non esistente o corrotto, ne creo uno nuovo
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
1
Inserisci il nome: re
Inserisci il indirizzo: novara
Inserisci lo stipendio: 1200
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
5
re, novara, 1200.000000, 1
Scegli una delle seguenti opzioni:
1. Inserisci
2. Cancella
3. Aggiorna stipendi
4. Stampa stipendi
5. Stampa contenuto archivio
6. Termina il programma
6
Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

FILE N° 3

Realizzare un programma che permetta all'utente, tramite un menù, di gestire un archivio di film. Dovranno essere disponibili le seguenti operazioni:

1. L'utente inserisce un certo numero di film:
 - 1.1.1. Codice identificativo numerico e titolo (senza spazi).
 - 1.1.2. I film dovranno essere inseriti con il codice già **ordinato** in senso crescente.
 - 1.1.3. L'inserimento della lista di film è terminato immettendo un codice pari a zero.
2. L'utente inserisce un codice e l'elaboratore verifica se il corrispondente film è presente nel file (ed in questo caso ne scrive i dati sul monitor).

Utilizzare l'algoritmo di ricerca binaria.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

inserisci

Dati d'input: *char nomeFile[]*, indica il nome del file

Dati in output:

Costanti: *N*, dimensione massima titolo film

Variabili di lavoro: **archivio*, puntatore al FILE, *nuovo Film* di tipo Film (*struct* contenente *codice e titolo* del film)

ordina

Dati d'input: *char nomeFile[]*, indica il nome del file

Dati di output:

Costanti: *N*, dimensione massima titolo film

Variabili di lavoro: **archivio*, puntatore al FILE, *filmCorrente* e *filmSuccessivo* di tipo Film, *unsigned int numFilm*, *i*, per spazzolare l'archivio dei film; *scambio* di tipo Booleano (*typedef enum {falso, vero} Booleano*)

ricerca

Dati d'input: *char nomeFile[]*, indica il nome del file

Dati in output:

Costanti: *N*, dimensione massima titolo film

Variabili di lavoro: *typedef enum {falso, vero} Booleano*; *Booleano trovato = falso*, **archivio*, puntatore al FILE; *film Corrente* di tipo Film; *unsigned int codCercato*; *inizio*, *fine*, *medio* di tipo intero per la ricerca binaria

menu

Dati d'input:

Dati in output: scelta fatta dal menù da ritornare al main

Costanti: *N*, dimensione massima titolo film

Variabili di lavoro: *scelta* di tipo unsigned int, per ritornare la scelta al main

main

Dati d'input:

Dati di output:

Costanti: *N*, dimensione massima titolo film

Variabili di lavoro: *nomeArchivio* indica il nome del file, *n* contiene il valore di menu, ovvero la scelta delle operazioni consentite sul file.

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
#define N 100
```

```

typedef struct
{
    unsigned int codice;
    char titolo[N];
} Film;
void ordina (char nomeFile[]);
void inserisci (char nomeFile[]);
void cerca (char nomeFile[]);
unsigned int menu ();
int main(void)
{
    char nomeArchivio[] = "numeri.dat";
    unsigned int n;
    do
    {
        n = menu();
        switch (n)
        {
            case 1:    inserisci (nomeArchivio);
                      break;
            case 2:    cerca (nomeArchivio);
                      break;
        }
    } while (n != 3);
    system ("pause");
    return(0);
}

void ordina (char nomeFile[])
{
    typedef enum {falso, vero} Booleano;
    FILE *archivio;
    Film filmCorrente, filmSuccessivo;
    unsigned int numFilm, i;
    Booleano scambio;
    /* "r+b" permette di leggere/scrivere su
    file binario esistente */
    archivio = fopen (nomeFile, "r+b");
    fseek (archivio, -sizeof(Film), SEEK_END);
    /* calcola numero di film inseriti nel file */
    numFilm = ftell (archivio) / sizeof(Film) + 1 ;
    do
    {
        scambio = falso;
        /* si riporta a inizio file */
        fseek (archivio, 0, SEEK_SET);
        for (i = 0; i < numFilm - 1; i++)
        {
            fread (&filmCorrente, sizeof(Film), 1, archivio);
            fread (&filmSuccessivo, sizeof(Film), 1, archivio);
            if (filmCorrente.codice > filmSuccessivo.codice)
            {
                /* sposta testina due record all'indietro */
                fseek (archivio, -2 * sizeof(Film), SEEK_CUR);
            }
        }
    }
}

```

```

        fwrite (&filmSuccessivo, sizeof(Film), 1,
archivio);
        fwrite (&filmCorrente, sizeof(Film), 1,
archivio);
        scambio = vero;
    }
}
} while (scambio);
fclose (archivio);
}
void inserisci (char nomeFile[])
{
FILE *archivio;
    Film nuovoFilm;

    /* apertura file per scrivere in append */
    archivio = fopen (nomeFile, "ab");
    printf ("Codice film diverso da 0 (0 per terminare inserimento): ");
    scanf ("%u", &nuovoFilm.codice);
    if (nuovoFilm.codice != 0)
    {
        printf ("Titolo film (le parole separale con_): ");
        scanf ("%s", nuovoFilm.titolo);
    }
    while (nuovoFilm.codice != 0)
    {
        fwrite (&nuovoFilm, sizeof(nuovoFilm), 1, archivio);
        printf ("Codice film diverso da 0: ");
        scanf ("%u", &nuovoFilm.codice);
        if (nuovoFilm.codice != 0)
        {
            printf ("Titolo film: ");
            scanf ("%s", nuovoFilm.titolo);
        }
    }
    fclose (archivio);
    ordina (nomeFile); /* riordina tutto il file */
}
void cerca (char nomeFile[])
{
    typedef enum {falso, vero} Booleano;
    Booleano trovato = falso;
    FILE *archivio;
    Film filmCorrente;
    unsigned int codCercato;
    int inizio, fine, medio;
    printf ("Codice cercato: ");
    scanf ("%u", &codCercato);
    archivio = fopen (nomeFile, "rb");
    inizio = 0;
    fseek (archivio, -sizeof(Film), SEEK_END);
    fine = ftell (archivio) / sizeof(Film);
    do
    {

```

```

    medio = (inizio + fine) / 2;
    fseek (archivio, medio * sizeof(Film), SEEK_SET);
    fread (&filmCorrente, sizeof(filmCorrente), 1,
    archivio);
    if (codCercato < filmCorrente.codice)
    {
        fine = medio - 1;
    }
    else if (codCercato > filmCorrente.codice)
    {
        inizio = medio + 1;
    }
    else
    {
        trovato = vero;
    }
} while (!trovato && inizio <= fine);
if (trovato)
{
    printf ("Trovato in posiz: %u\n", medio);
    printf ("Codice: %u\n", filmCorrente.codice);
    printf ("Titolo: %s\n", filmCorrente.titolo);
}
else
{
    printf ("Non trovato!\n");
}
fclose (archivio);
}

unsigned int menu ()
{
    unsigned int scelta;
    printf ("1. Inserisci elenco film\n");
    printf ("2. Cerca film\n");
    printf ("3. Esci\n");
    printf ("\nScelta: ");
    scanf ("%u", &scelta);
    return (scelta);
}

```

```
C:\Dev-Cpp\file3dat.exe
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 2
Codice cercato: 2
Non trovato!
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 1
Codice film diverso da 0 <0 per terminare inserimento>: 1
Titolo film <le parole separale con_>: rambo
Codice film diverso da 0: 2
Titolo film: Rockyl
Codice film diverso da 0: 0
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 2
Codice cercato: 1
Trovato in posiz: 3
Codice: 1
Titolo: rambo
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 3
Premere un tasto per continuare . . .
```

Microsoft Visual C++

```
#include <stdio.h>
#include <stdlib.h>
#define N 100
typedef struct
{
    unsigned int codice;
    char titolo[N];
} Film;
void ordina (char nomeFile[]);
void inserisci (char nomeFile[]);
void cerca (char nomeFile[]);
unsigned int menu ();
int main(void)
{
    char nomeArchivio[] = "numeri.dat";
    unsigned int n;
    do
    {
        n = menu();
        switch (n)
        {
            case 1:    inserisci (nomeArchivio);
                       break;
            case 2:    cerca (nomeArchivio);
                       break;
        }
    } while (n != 3); system ("pause");return(0);
}
void ordina (char nomeFile[])
```

```

{
    typedef enum {falso, vero} Booleano;
    FILE *archivio;
    Film filmCorrente, filmSuccessivo;
    unsigned int numFilm, i;
    Booleano scambio;
    /* "r+b" permette di leggere/scrivere su file binario esistente */
    archivio = fopen (nomeFile, "r+b");
    fseek (archivio, -sizeof(Film), SEEK_END);
    /* calcola numero di film inseriti nel file */
    numFilm = ftell (archivio) / sizeof(Film) + 1 ;
    do
    {
        scambio = falso;
        /* si riporta a inizio file */
        fseek (archivio, 0, SEEK_SET);
        for (i = 0; i < numFilm - 1; i++)
        {
            fread (&filmCorrente, sizeof(Film), 1, archivio);
            fread (&filmSuccessivo, sizeof(Film), 1, archivio);
            if (filmCorrente.codice > filmSuccessivo.codice)
            {
                /* sposta testina due record all'indietro */
                fseek (archivio, -2 * sizeof(Film), SEEK_CUR);
                fwrite (&filmSuccessivo, sizeof(Film), 1,
                    archivio);
                fwrite (&filmCorrente, sizeof(Film), 1,
                    archivio);
                scambio = vero;
            }
        }
    } while (scambio);
    fclose (archivio);
}

void inserisci (char nomeFile[])
{
    FILE *archivio;
    Film nuovoFilm;

    /* apertura file per scrivere in append */
    archivio = fopen (nomeFile, "ab");
    printf ("Codice film diverso da 0 (0 per terminare inserimento): ");
    scanf ("%u", &nuovoFilm.codice);
    if (nuovoFilm.codice != 0)
    {
        printf ("Titolo film (le parole separate con _): ");
        scanf ("%s", nuovoFilm.titolo);
    }
    while (nuovoFilm.codice != 0)
    {
        fwrite (&nuovoFilm, sizeof(nuovoFilm), 1, archivio);
        printf ("Codice film diverso da 0: ");
        scanf ("%u", &nuovoFilm.codice);
        if (nuovoFilm.codice != 0)

```

```

        {
            printf ("Titolo film: ");
            scanf ("%s", nuovoFilm.titolo);
        }
    }
    fclose (archivio);
    ordina (nomeFile); /* riordina tutto il file */
}
void cerca (char nomeFile[])
{
    typedef enum {falso, vero} Booleano;
    Booleano trovato = falso;
    FILE *archivio;
    Film filmCorrente;
    unsigned int codCercato;
    int inizio, fine, medio;
    printf ("Codice cercato: ");
    scanf ("%u", &codCercato);
    archivio = fopen (nomeFile, "rb");
    inizio = 0;
    fseek (archivio, -sizeof(Film), SEEK_END);
    fine = ftell (archivio) / sizeof(Film);
    do
    {
        medio = (inizio + fine) / 2;
        fseek (archivio, medio * sizeof(Film), SEEK_SET);
        fread (&filmCorrente, sizeof(filmCorrente), 1,
            archivio);
        if (codCercato < filmCorrente.codice)
        {
            fine = medio - 1;
        }
        else if (codCercato > filmCorrente.codice)
        {
            inizio = medio + 1;
        }
    }
    else
    {
        trovato = vero;
    }
    while (!trovato && inizio <= fine);
    if (trovato)
    {
        printf ("Trovato in posiz: %u\n", medio);
        printf ("Codice: %u\n", filmCorrente.codice);
        printf ("Titolo: %s\n", filmCorrente.titolo);
    }
    else
    {
        printf ("Non trovato!\n");
    }
    fclose (archivio);
}
unsigned int menu ()

```

```

{
    unsigned int scelta;
    printf ("1. Inserisci elenco film\n");
    printf ("2. Cerca film\n");
    printf ("3. Esci\n");
    printf ("\nScelta: ");
    scanf ("%u", &scelta);
    return (scelta);
}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\file3bina\Debug\file3bina.exe
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 1
Codice film diverso da 0 <0 per terminare inserimento>: 1
Titolo film <le parole separale con_>: rambo
Codice film diverso da 0: 3
Titolo film: re
Codice film diverso da 0: 0
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 2
Codice cercato: 3
Trovato in posiz: 5
Codice: 3
Titolo: re
1. Inserisci elenco film
2. Cerca film
3. Esci

Scelta: 3
Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

OOP N° 1

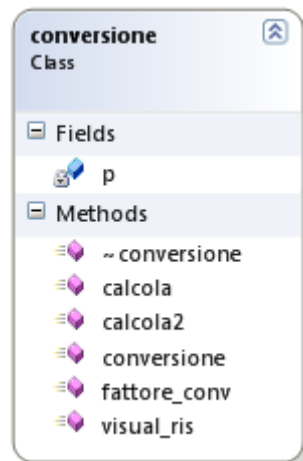
Definire una classe convertitore, comprensiva di costruttore e distruttore, al fine di realizzare un convertitore di valute, ad esempio lire/euro. Per il programma si richiede la scrittura di almeno una funzione membro di calcolo, una di output e la definizione di uno o più campi privati ove memorizzare i dati, nonché la scrittura del main

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream.h>
#include <stdio.h>
#include <stdlib.h>
using namespace std;
// definizione classe
class conversione
{
    public:
        conversione(); //costruttore
        ~conversione(); //distruttore
        void fattore_conv(float fatt); //funz. di set fattore di conversione
        void calcola(float num); //funzione di calcolo euro/lire
        void calcola2(float num); //funzione di calcolo lire/euro
        float visual_ris(); //funzione di interfaccia
    private:
        float *p;
};
// implementazione classe
conversione::conversione() {
    p = new float [2];
    p[0]=0; //fattore di conversione
    p[1]=0; //risultato della conversione
}
```

```

conversione::~conversione() {
    delete[]p;}
void conversione::fattore_conv(float fatt){
    p[0]=fatt;}
void conversione::calcola(float num){
    p[1] = num / p[0];}
void conversione::calcola2(float num){
    p[1] = num * p[0];}
float conversione::visual_ris(){
    return p[1];}
int main(void){
    int scelta, r;
    float fatt,num;
    conversione A;
    r=1;
    while(r){
        cout<<"\n ***** EURO-CONVERTITORE *****\n";
        cout<<"\n *****",
        cout<<"\n * per la conversione lire/euro digita 1 *",
        cout<<"\n * per la conversione euro/lire digita 2 *",
        cout<<"\n * per un altro tipo di conver. digita 3 *",
        cout<<"\n * per uscire digita 4 *****",
        cout<<"\n *****",
        cout<<"\n Fai la tua scelta: ";
        cin>>scelta;
        switch(scelta){
            case 1: A.fattore_conv(1936.27);
                cout<<"\n Inserisci la cifra da convertire: ";
                cin>>num;
                A.calcola(num);
                cout<<" Il risultato della conversione e' "<<A.visual_ris()<<endl;
                system("PAUSE");
                break;
            case 2: A.fattore_conv(1936.27);
                cout<<"\n Inserisci la cifra da convertire: ";
                cin>>num;
                A.calcola2(num);
                cout<<"\n Il risultato della conversione e'"<<A.visual_ris()<<endl;
                system("PAUSE");
                break;
            case 3: cout<<"\n Inserisci il fattore di conversione: ";
                cin>>fatt;
                A.fattore_conv(fatt);
                cout<<"\n Inserisci la cifra da convertire: ";
                cin>>num;
                A.calcola(num);
                cout<<"\n Il risultato della conversione e'"<<A.visual_ris()<<endl;
                system("PAUSE");
                break;
            case 4: r=0;
                cout<<"\n Arrivederci al prossimo utilizzo!";
                break;
            default: cout<<"\n <Scelta non valida!>";
                break;}
    }
}

```

```

        cout<<"\n\n\n\n\n";}
system("pause");
return(0);}

```

```

C:\Dev-Cpp\loop1.exe
***** EURO-CONVERTITORE *****
*****
* per la conversione lire/euro digita 1 *
* per la conversione euro/lire digita 2 *
* per un altro tipo di conver. digita 3 *
* per uscire digita 4 *****
*****
Fai la tua scelta: 1

Inserisci la cifra da convertire: 1234
Il risultato della conversione e' 0.637308
Premere un tasto per continuare . . . _

```

Microsoft Visual C++

```

#include <iostream>
#include <stdio.h>
using namespace std;
// definizione classe
class conversione
{
    public:
        conversione(); //costruttore
        ~conversione(); //distruttore
        void fattore_conv(float fatt); //funz. di set fattore di conversione
        void calcola(float num); //funzione di calcolo euro/lire
        void calcola2(float num); //funzione di calcolo lire/euro
        float visual_ris(); //funzione di interfaccia
    private:
        float *p;
};
// implementazione classe
conversione::conversione() {
    p = new float [2];
    p[0]=0; //fattore di conversione
    p[1]=0; //risultato della conversione
}
conversione::~~conversione() {
    delete[]p;}
void conversione::fattore_conv(float fatt){
    p[0]=fatt;}
void conversione::calcola(float num){
    p[1] = num / p[0];}
void conversione::calcola2(float num){
    p[1] = num * p[0];}
float conversione::visual_ris(){
    return p[1];}
int main(void){
    int scelta, r;
    float fatt,num;
    conversione A;

```

```

r=1;
while(r){
cout<<"\n ***** EURO-CONVERTITORE *****\n";
cout<<"\n *****",
cout<<"\n * per la conversione lire/euro digita 1 *",
cout<<"\n * per la conversione euro/lire digita 2 *",
cout<<"\n * per un altro tipo di conver. digita 3 *",
cout<<"\n * per uscire digita 4 *****",
cout<<"\n *****",
cout<<"\n Fai la tua scelta: ";
cin>>scelta;
switch(scelta){
    case 1: A.fattore_conv(1936.27);
            cout<<"\n Inserisci la cifra da convertire: ";
            cin>>num;
            A.calcola(num);
            cout<<" Il risultato della conversione e' "<<A.visual_ris()<<endl;
            //system("pause");
            break;
    case 2: A.fattore_conv(1936.27);
            cout<<"\n Inserisci la cifra da convertire: ";
            cin>>num;
            A.calcola2(num);
            cout<<"\n Il risultato della conversione e'"<<A.visual_ris()<<endl;
            //system("pause");
            break;
    case 3: cout<<"\n Inserisci il fattore di conversione: ";
            cin>>fatt;
            A.fattore_conv(fatt);
            cout<<"\n Inserisci la cifra da convertire: ";
            cin>>num;
            A.calcola(num);
            cout<<"\n Il risultato della conversione e'"<<A.visual_ris()<<endl;
            //system("PAUSE");
            break;
    case 4: r=0;
            cout<<"\n Arrivederci al prossimo utilizzo!";
            break;
    default: cout<<"\n <Scelta non valida!>";
            break;}
    cout<<"\n\n\n\n\n";}
return(0);}

```

```
c:\documents and settings\hp\documenti\visual studio 2010\Projects\oop1\Debug\oop1.exe

*****
* per la conversione lire/euro digita 1 *
* per la conversione euro/lire digita 2 *
* per un altro tipo di conver. digita 3 *
* per uscire digita 4 *****
*****
Fai la tua scelta: 1

Inserisci la cifra da convertire: 1500
Il risultato della conversione e' 0.774685

***** EURO-CONVERTITORE *****
*****
* per la conversione lire/euro digita 1 *
* per la conversione euro/lire digita 2 *
* per un altro tipo di conver. digita 3 *
* per uscire digita 4 *****
*****
Fai la tua scelta: 
```

5. Documentazione

6. Testing

OOP N° 2

Definire la classe Calcolatrice che possa svolgere le quattro operazioni elementari.

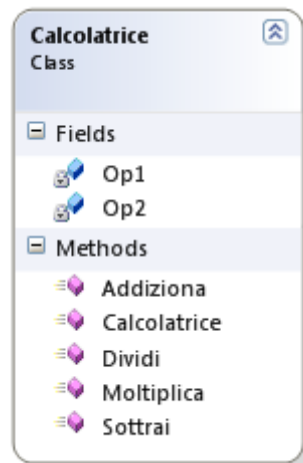
1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Le uniche variabili membro (private) sono i due operandi Op1 e Op2. Definiamo quattro funzioni membro, una per ogni operazione elementare. Dal programma principale (che occorrerà sviluppare separatamente) si propone dapprima la scelta dell'operazione che si vuole eseguire, poi si richiede l'inserimento dei due operandi.

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream.h>
using namespace std;
class Calcolatrice{
private:
double Op1;
double Op2;
public:
Calcolatrice( ) {
Op1=0
Op2=0;} // fine costruttore
double Addiziona( ){
return (Op1+Op2);}
double Sottrai( ){
return (Op1-Op2);}
double Moltiplica( ){
return (Op1*Op2);}
double Dividi( ) {
return (Op1/Op2);}}
```

Microsoft Visual C++

```
#include <iostream>
using namespace std;
class Calcolatrice{
```

```
private:
double Op1;
double Op2;
public:
Calcolatrice( ) {
Op1=0
Op2=0;} // fine costruttore
double Addiziona( ){
return (Op1+Op2);}
double Sottrai( ){
return (Op1-Op2);}
double Moltiplica( ){
return (Op1*Op2);}
double Dividi( ) {
return (Op1/Op2);}}
```

5. Documentazione

6. Testing

OOP N° 3

Definire la classe Vettore come classe generica che possa caricare, visualizzare e sommare le componenti del vettore.

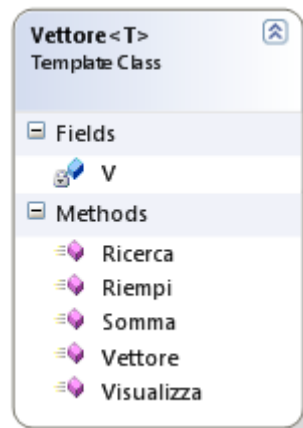
1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Come possiamo osservare, analogamente a quanto visto per le funzioni generiche, il tipo T è utilizzato come tipo parametrico, nel senso che è un segnaposto per il tipo reale che verrà passato come parametro al momento della creazione di un oggetto di classe Vettore.

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream.h>
using namespace std;
const int MAXEL=10;
template <class T>
class Vettore {
private:
    T* V; // variabili istanza
public:
    Vettore( );
    void Riempi( );
    void Visualizza( );
    bool Ricerca(T Num);
    T Somma( );}; // fine classe Vettore
template <class T>
Vettore<T>::Vettore( ) {
    V = new T[MAXEL]; }
template <class T>
void Vettore<T>::Riempi( ) {
    int l=0;
    while (l<MAXEL) {
        cout << "Inserire il nuovo elemento:"; cin >> V[l];
        l++;} // fine Riempi
template <class T>
void Vettore<T>::Visualizza( ) {
```



```

int l=0;
while (l<MAXEL) {
cout << l << " elemento: " << V[l] << '\n';
l = l + 1; } }
template <class T>
bool Vettore<T>::Ricerca(T Num) {
int l=0;
while (l<MAXEL) {
if (V[l] == Num)
return(true);
l = l + 1;}
return(false);}
template <class T>
T Vettore<T>::Somma( ) {
T S=0;
int l=0;
while (l<MAXEL) {
S = S + V[l];
l = l + 1;}
return(S);}

```

Microsoft Visual C++

```

#include <iostream>
using namespace std;
const int MAXEL=10;
template <class T>
class Vettore {
private:
T* V; // variabili istanza
public:
Vettore( );
void Riempi( );
void Visualizza( );
bool Ricerca(T Num);
T Somma( );}; // fine classe Vettore
template <class T>
Vettore<T>::Vettore( ) {
V = new T[MAXEL]; }
template <class T>
void Vettore<T>::Riempi( ) {
int l=0;
while (l<MAXEL) {
cout << "Inserire il nuovo elemento: "; cin >> V[l];
l++;}} // fine Riempi
template <class T>
void Vettore<T>::Visualizza( ) {
int l=0;
while (l<MAXEL) {
cout << l << " elemento: " << V[l] << '\n';
l = l + 1; } }
template <class T>
bool Vettore<T>::Ricerca(T Num) {
int l=0;
while (l<MAXEL) {

```

```
if (V[l] == Num)
return(true);
l = l + 1;}
return(false);}
template <class T>
T Vettore<T>::Somma( ) {
T S=0;
int l=0;
while (l<MAXEL) {
S = S + V[l];
l = l + 1;}
return(S);}

```

5. Documentazione

6. Testing

OOP N° 4

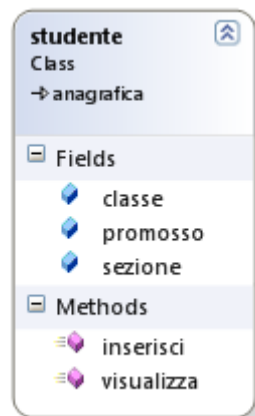
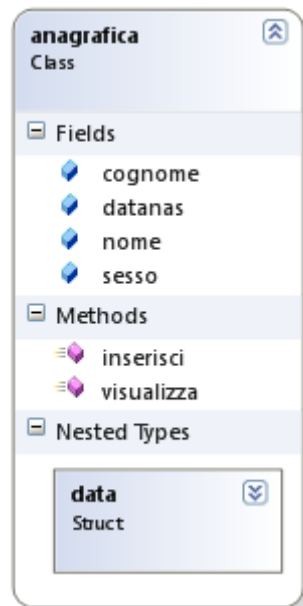
Definire la classe base anagrafica contenente nome, cognome, data di nascita (giorno, mese, anno), sesso con i metodi per inserimento e visualizzazione dei dati. Definire poi la classe derivata studente, contenente anche la classe, la sezione e il campo promosso (tipo bool). Definire anche il main.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream>
using namespace std;
```

```
/* Programma Esempio di ereditarietà anagrafica= classe base; studente= classe derivata*/
```

```
class anagrafica { // classe base
    struct data { short int giorno, mese, anno;};
```

```

public:
string cognome;
string nome;
data datanas;
char sesso;
void inserisci(){
    cout << "Inserisci il Cognome: "; cin >> cognome ;
    cout << "Inserisci il Nome: "; cin >> nome;
    cout << "Inserisci la data di nascita nel formato gg mm aa: ";
    cin>>datanas.giorno>> datanas.mese>> datanas.anno;
    cout <<"Inserisci il sesso: M/F "; cin >> sesso; }
void visualizza(){
    cout <<cognome<<"\t"<<nome<<"\t"
    << datanas.giorno<<"\t"<<datanas.mese<<"\t"<<datanas.anno
    <<"\t"<< sesso;
} };
class studente : public anagrafica { // studente classe derivata
public:
short int classe;
char sezione;
bool promosso;
// overriding dei metodi
void inserisci(){
    anagrafica::inserisci();
    cout << "Inserisci la classe: "; cin >> classe;
    cout << "Inserisci la sezione: "; cin >> sezione;
    char risp;
    cout << "E' stato promosso? (S/N) "; cin >> risp;
    if (risp=='S') promosso=true; else promosso =false;
}
void visualizza(){
    anagrafica::visualizza();
    cout << "\t"<< classe<< sezione<<' ';
    if (promosso) cout <<"Promosso"; else cout<<"Respinto";
}
};
int main(void) {
    anagrafica a; studente s;
    cout <<"Spazio occupato da anagrafica "<<sizeof(a)<<endl;
    cout <<"Spazio occupato da studente "<<sizeof(s)<<endl;
    a.inserisci();
    a.visualizza();
    cout << endl;
    s.inserisci();
    s.visualizza();
    cout << endl;
    system("Pause");
    return (0);
}

```

```
C:\Dev-Cpp\loop5.exe
Spazio occupato da anagrafica 16
Spazio occupato da studente 20
Inserisci il Cognome: re
Inserisci il Nome: mario
Inserisci la data di nascita nel formato gg mm aa: 3 6 98
Inserisci il sesso: M/F M
re mario 3\6\98 M
Inserisci il Cognome: reco
Inserisci il Nome: marco
Inserisci la data di nascita nel formato gg mm aa: 23 11 99
Inserisci il sesso: M/F M
Inserisci la classe: 2
Inserisci la sezione: B
E' stato promosso? (S/N) S
reco marco 23\11\99 M 2B Promosso
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

```
#include <iostream>
#include<string>
using namespace std;
/* Programma Esempio di ereditarietà anagrafica= classe base; studente= classe
derivata*/
class anagrafica { // classe base
    struct data { short int giorno, mese, anno;};
    public:
    string cognome;
    string nome;
    data datanas;
    char sesso;
    void inserisci(){
        cout << "Inserisci il Cognome: "; cin >> cognome ;
        cout << "Inserisci il Nome: "; cin >> nome;
        cout << "Inserisci la data di nascita nel formato gg mm aa: ";
        cin>>datanas.giorno>> datanas.mese>> datanas.anno;
        cout <<"Inserisci il sesso: M/F "; cin >> sesso; }
    void visualizza(){
        cout <<cognome<<"\t"<<nome<<"\t"
        << datanas.giorno<<"\\"<<datanas.mese<<"\\"<<datanas.anno
        <<"\t"<< sesso;
    }
};
class studente : public anagrafica { // studente classe derivata
    public:
    short int classe;
    char sezione;
    bool promosso;
    // overriding dei metodi
    void inserisci(){
        anagrafica::inserisci();
        cout << "Inserisci la classe: "; cin >> classe;
        cout << "Inserisci la sezione: "; cin >> sezione;
        char risp;
        cout << "E' stato promosso? (S/N) "; cin >> risp;
        if (risp=='S') promosso=true; else promosso =false;
    }
    void visualizza(){
        anagrafica::visualizza();
```

```

    cout << '\t' << classe << sezione << ' ';
    if (promosso) cout << "Promosso"; else cout << "Respinto";
}

```

```

C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\oop4\Debug\oop4.exe
Spazio occupato da anagrafica 72
Spazio occupato da studente 76
Inserisci il Cognome: re
Inserisci il Nome: mario
Inserisci la data di nascita nel formato gg mm aa: 23 4 97
Inserisci il sesso: M/F M
re      mario      23\4\97 M
Inserisci il Cognome: rex
Inserisci il Nome: maria
Inserisci la data di nascita nel formato gg mm aa: 23 6 98
Inserisci il sesso: M/F F
Inserisci la classe: 3
Inserisci la sezione: A
E' stato promosso? (S/N) S
rex      maria      23\6\98 F      3A Promosso
Premere un tasto per continuare . . .

```

5. Documentazione

6. Testing

OOP N° 5

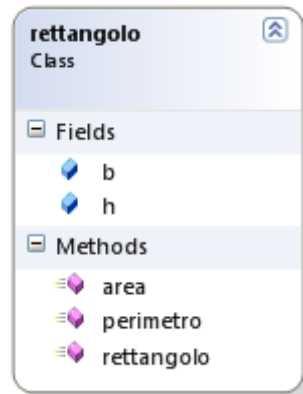
Definire la classe Rettangolo che permetta di calcolare l'area e il perimetro di un rettangolo ricevendo in input base e altezza. Definire anche il main. Fornire l'output per due rettangoli

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream>
using namespace std;
class rettangolo {
    // attributi
public:
    float b,h;
    // metodi
    rettangolo(float x=1.0, float y=1.0) {b=x; h=y;} // costruttore
    float area(){return b*h;}
    float perimetro() {return 2*(b+h);}
};

int main(void) {
    rettangolo r1(2,5), r2;
    r2=r1; // costruttore per copia richiamato automaticamente dall'operatore=
    r2.b=3; // cambio solo la base
    cout << "Base= " << r1.b << "\tAltezza= " << r1.h << "\tArea = " << r1.area() << "\tPerimetro
= " << r1.perimetro() << endl;
    cout << "Base= " << r2.b << "\tAltezza= " << r2.h << "\tArea = " << r2.area() << "\tPerimetro
= " << r2.perimetro() << endl;
    system("Pause");
    return (0);
}
```

```
C:\Dev-Cpp\loop4.exe
Base= 2 Altezza= 5      Area = 10      Perimetro = 14
Base= 3 Altezza= 5      Area = 15      Perimetro = 16
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

```
#include <iostream>
using namespace std;
class rettangolo {
    // attributi
public:
    float b,h;
    // metodi
    rettangolo(float x=1.0, float y=1.0) {b=x; h=y;} // costruttore
    float area(){return b*h;}
    float perimetro() {return 2*(b+h);}
};

int main(void) {
    rettangolo r1(2,5), r2;
    r2=r1; // costruttore per copia richiamato automaticamente dall'operatore=
    r2.b=3; // cambio solo la base
    cout << "Base= " << r1.b << "\tAltezza= " << r1.h << "\tArea = " << r1.area() << "\tPerimetro
= " << r1.perimetro() << endl;
    cout << "Base= " << r2.b << "\tAltezza= " << r2.h << "\tArea = " << r2.area() << "\tPerimetro
= " << r2.perimetro() << endl;
    system("Pause");
    return (0);
}
```

```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\loop5\Debug\loop5.exe
Base= 2 Altezza= 5      Area = 10      Perimetro = 14
Base= 3 Altezza= 5      Area = 15      Perimetro = 16
Premere un tasto per continuare . . . _
```

5. Documentazione

6. Testing

OOP N° 6

Scrivere un programma che utilizzi i metodi degli oggetti string: acquisizione due stringhe, confronto alfabetico, lunghezze, sizeof, inserimento di caratteri in mezzo a una stringa, ricerca di sottostringhe 'la', estrazioni di sottostringhe, sostituzioni di pezzi di stringhe.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello

4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream>
using namespace std;
/* Programma metodi degli oggetti string */
int main(void) {
    string a,b,s,s1;
    cout << "Inserisci la prima stringa: ";getline(cin,a);
    cout << "Inserisci la seconda stringa. "; getline(cin,b);

    cout << "SizeOf(prima stringa) "<< sizeof(a)<< "\t SizeOF(seconda stringa):
"<< sizeof(b)<< endl;
    cout << "Metodo length()= lunghezza di una stringa. "<< endl;
    cout << "Lunghezza della stringa "<< a<< " = "<< a.length()<< endl;
    cout << "Lunghezza della stringa "<< b<< " = "<< b.size()<< endl;
    // confronto tra due stringhe
    if (a==b) cout << "\"<a << " e' uguale a " << b << "\" << endl;
    else if (a<b) cout << "\"<a << " e' minore di " << b << "\" << endl;
    else cout << "\"<a << " e' maggiore di " << b << "\" << endl;
    // somma di due stringhe
    s= a+ " "+b; cout<< "\nSomma delle stringhe e di uno spazio: "<<s<<endl;
    // inserimento di caratteri in mezzo ad una stringa INSERT
    s=s.insert(2,"Riminese ");
    cout<< "\nInserimento di caratteri in mezzo: insert() " << s<< "\"<<endl;

    cout << "\nRicerca di sottostringhe: find() "<< endl;
    int pp;
    pp=a.find("la"); cout << "In "<a<< " 'la' si trova nella posizione "<< pp<< endl;
    pp=b.find("la"); cout << "In "<b<< " 'la' si trova nella posizione "<< pp<< endl;

    cout << "\nEstrazione di sottostringhe substr() "<< endl;
    cout << a.substr(0,3)+b.substr(0,3)<< "\t "<a.substr(4)<< endl;

    cout << "\nSostituzioni di pezzi di stringa: replace() "<< endl;
    s1=a; s1.replace(0,5,"Ferra"); cout << s1<< endl;
    system("Pause");
    return (0);
}
```

```
C:\Documents and Settings\HP\Documenti\oop\oop6.exe
Inserisci la prima stringa: laura
Inserisci la seconda stringa. anna
SizeOf(prima stringa) 4 SizeOf(seconda stringa): 4
Metodo length()= lunghezza di una stringa.
Lunghezza della stringa 'laura'= 5
Lunghezza della stringa 'anna'= 4
'laura' e' maggiore di 'anna'

Somma delle stringhe e di uno spazio: laura anna

Inserimento di caratteri in mezzo: insert() 'laRiminese ura anna'

Ricerca di sottostringhe: find()
In laura 'la' si trova nella posizione 0
In anna 'la' si trova nella posizione -1

Estrazione di sottostringhe substr()
lauann a

Sostituzioni di pezzi di stringa: replace()
Ferra
Premere un tasto per continuare . . . _
```

Microsoft Visual C++

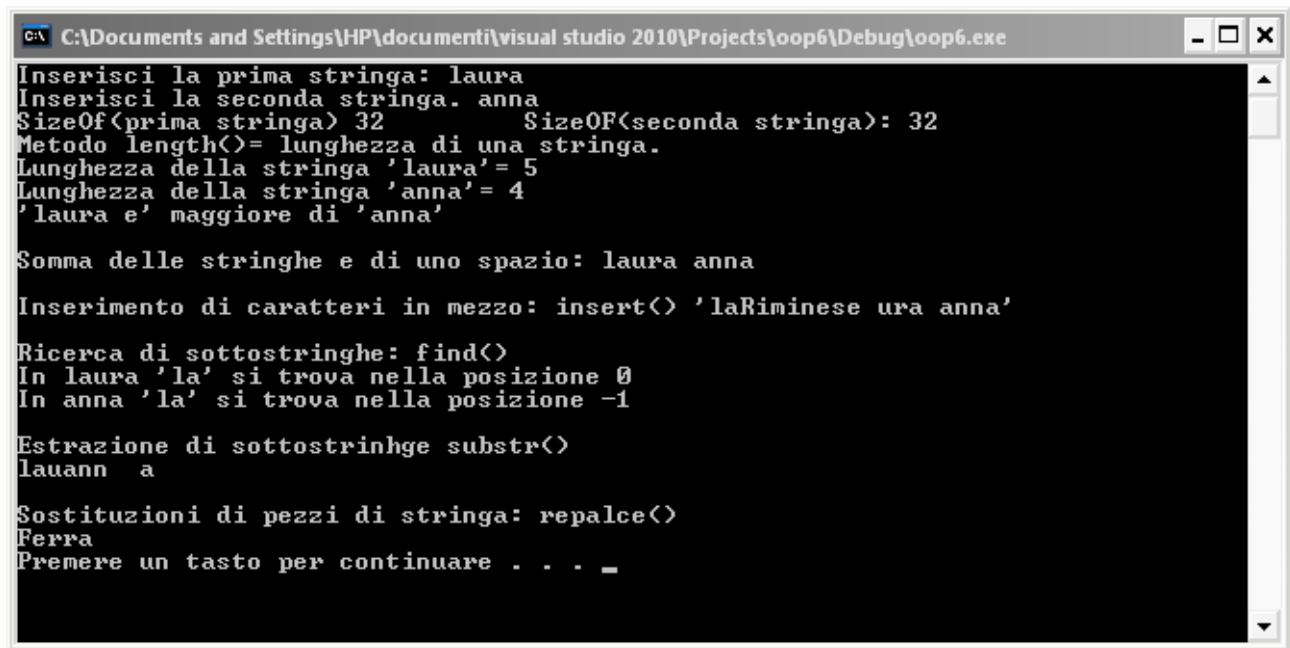
```
#include <iostream>
#include<string>
using namespace std;
/* Programma metodi degli oggetti string*/
int main(void) {
    string a,b,s,s1;
    cout << "Inserisci la prima stringa: "; getline(cin,a);
    cout << "Inserisci la seconda stringa. "; getline(cin,b);
    cout << "SizeOf(prima stringa) "<< sizeof(a)<< "\t SizeOf(seconda stringa): "
    "<<sizeof(b)<<endl;
    cout <<"Metodo length()= lunghezza di una stringa. "<<endl;
    cout <<"Lunghezza della stringa '"<< a<< "'=" "<<a.length()<< endl;
    cout <<"Lunghezza della stringa '"<< b<< "'=" "<<b.size()<< endl;
    // confronto tra due stringhe
    if (a==b) cout << "\"<<a << " e' uguale a '" << b <<"\"<<endl;
    else if (a<b) cout << "\"<<a << " e' minore di '" << b <<"\"<<endl;
    else cout << "\"<<a << " e' maggiore di '" << b <<"\"<<endl;
    // somma di due stringhe
    s=a+" "+b; cout<< "\nSomma delle stringhe e di uno spazio: "<<s<<endl;
    // inserimento di caratteri in mezzo ad una stringa INSERT
    s=s.insert(2,"Riminese ");
    cout<< "\nInserimento di caratteri in mezzo: insert() '" << s<<"\"<<endl;

    cout << "\nRicerca di sottostringhe: find() "<< endl;
    int pp;
    pp=a.find("la"); cout << "In '"<<a<<" 'la' si trova nella posizione '"<< pp<< endl;
    pp=b.find("la"); cout << "In '"<<b<<" 'la' si trova nella posizione '"<< pp<< endl;

    cout << "\nEstrazione di sottostringhe substr() "<<endl;
    cout << a.substr(0,3)+b.substr(0,3)<< "\t "<<a.substr(4)<<endl;

    cout << "\nSostituzioni di pezzi di stringa: replace() "<< endl;
    s1=a; s1.replace(0,5,"Ferra"); cout <<s1<<endl;
    system("Pause");
}
```

```
    return (0);  
}
```



The screenshot shows a Windows command prompt window titled "C:\Documents and Settings\HP\documenti\visual studio 2010\Projects\oop6\Debug\oop6.exe". The window contains the following text:

```
Inserisci la prima stringa: laura  
Inserisci la seconda stringa. anna  
SizeOf(prima stringa) 32      SizeOf(seconda stringa): 32  
Metodo length()= lunghezza di una stringa.  
Lunghezza della stringa 'laura'= 5  
Lunghezza della stringa 'anna'= 4  
'laura e' maggiore di 'anna'  
  
Somma delle stringhe e di uno spazio: laura anna  
  
Inserimento di caratteri in mezzo: insert() 'laRiminense ura anna'  
  
Ricerca di sottostringhe: find()  
In laura 'la' si trova nella posizione 0  
In anna 'la' si trova nella posizione -1  
  
Estrazione di sottostringhe substr()  
lauann a  
  
Sostituzioni di pezzi di stringa: repalce()  
Ferra  
Premere un tasto per continuare . . . _
```

5. Documentazione

6. Testing

OOP N° 7

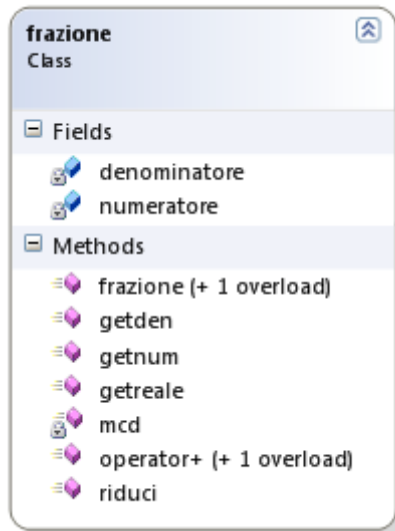
Definire la classe frazione che possa ricevere in input numeratore e denominatore, ridurre ai minimi termini la frazione, sommare due frazioni. Definire anche il main

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <cmath>
#include <iostream>
using namespace std;
class frazione {
    int numeratore;
    int denominatore;
    int mcd(){
        // massimo comun divisore tra numeratore e denominatore
        int a=numeratore, b=denominatore, r;
        if (a<b){r=a; a=b;b=r;}
        r=a%b;
        while (r!=0) { a=b; b=r; r=a%b;}
        return b;
    }
public:
    int getnum() {return numeratore;}
    int getden() {return denominatore;}
    void riduci(){int m=mcd(); numeratore /=m; denominatore /=m;}
    frazione(int a=1, int b=1) // costruttore a partire da due numeri interi
    {numeratore=a; denominatore=b; }
    frazione(double a) { // overload del costruttore
        // costruttore a partire da un numero double; riduce anche ai minimi termini
        int i=0, n, d=1;
        while (trunc(a) != a && i < 9) {a*=10; i++; d*=10;}
```

```

    numeratore=trunc(a);
    denominatore=d;
    riduci();
}
double getreale() {return (double)numeratore / denominatore;}
frazione operator + (frazione f1){
    frazione f(f1.getnum()*denominatore+numeratore*f1.getden(),
f1.getden()*denominatore);
    f.riduci();
    return f;
}
frazione operator + (int n){ // overload dell'operatore + con un intero
    frazione f(n,1); // crea la frazione n/1
    return *this + f; // richiama l'operatore + originario
}
};
int main(void) {
    frazione fr1(1,4), fr2(0.75), fr3;

    fr3=fr1; // l'assegnazione è automaticamente definita con il costruttore per copia?

    cout <<"fr1= "<<fr1.getnum()<<"/"<<fr1.getden();
    cout <<"\tfr2= "<<fr2.getnum()<<"/"<<fr2.getden();
    cout <<"\tfr3= "<<fr3.getnum()<<"/"<<fr3.getden()<<endl;
    fr3=fr1+fr2;
    cout << "Dopo la somma fr3=fr1+fr2 ..."<<endl;
    cout <<"fr1= "<<fr1.getnum()<<"/"<<fr1.getden();
    cout <<"\tfr2= "<<fr2.getnum()<<"/"<<fr2.getden();
    cout <<"\tfr3= "<<fr3.getnum()<<"/"<<fr3.getden()<< " = "<< fr3.getreale()<<endl;
    fr3=fr1+5;
    cout << "Dopo la somma fr3=fr1+5 ..."<<endl;
    cout <<"fr1= "<<fr1.getnum()<<"/"<<fr1.getden();
    cout <<"\tfr2= "<<fr2.getnum()<<"/"<<fr2.getden();
    cout <<"\tfr3= "<<fr3.getnum()<<"/"<<fr3.getden()<< " = " <<fr3.getreale()<<endl;
    system("Pause");
    return 0;
}

```

```

C:\Dev-Cpp\loop7.exe
fr1= 1/4      fr2= 3/4      fr3= 1/4
Dopo la somma fr3=fr1+fr2 ...
fr1= 1/4      fr2= 3/4      fr3= 1/1 = 1
Dopo la somma fr3=fr1+5 ...
fr1= 1/4      fr2= 3/4      fr3= 21/4 = 5.25
Premere un tasto per continuare . . .

```

Microsoft Visual C++

```

#include <cmath.h>
#include <iostream>
using namespace std;
double trunc(double d){ return (d>0) ? floor(d) : ceil(d) ; }
using namespace std;
class frazione {
    int numeratore;

```

```

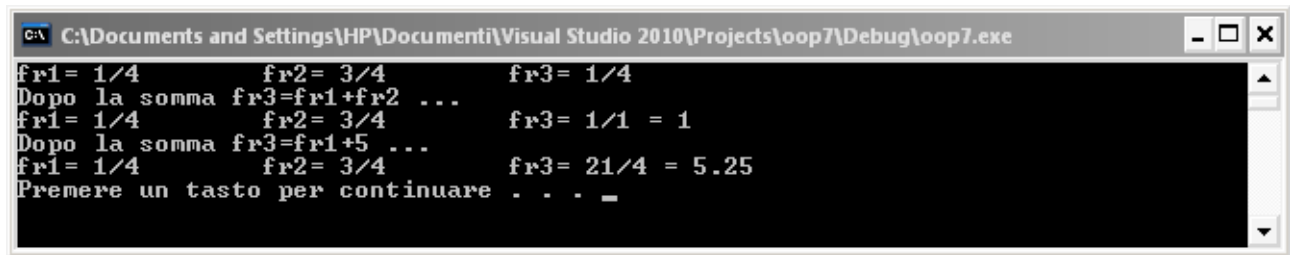
int denominatore;
int mcd(){
    // massimo comun divisore tra numeratore e denominatore
    int a=numeratore, b=denominatore,r;
    if (a<b){r=a; a=b;b=r;}
    r=a%b;
    while (r!=0) { a=b; b=r; r=a%b;}
    return b;
}
public:
int getnum() {return numeratore;}
int getden() {return denominatore;}
void riduci(){int m=mcd(); numeratore /=m; denominatore /=m;}
frazione(int a=1, int b=1) // costruttore a partire da due numeri interi
{numeratore=a; denominatore=b; }
frazione(double a) { // overload del costruttore
    // costruttore a partire da un numero double; riduce anche ai minimi termini
    int i=0, n, d=1;
    while (trunc(a) != a && i < 9) {a*=10; i++; d*=10;}
    numeratore=trunc(a);
    denominatore=d;
    riduci();
}
double getreale() {return (double)numeratore / denominatore;}
frazione operator + (frazione f1){
    frazione f(f1.getnum()*denominatore+numeratore*f1.getden(),
f1.getden()*denominatore);
    f.riduci();
    return f;
}
frazione operator + (int n){ // overload dell'operatore + con un intero
    frazione f(n,1); // crea la frazione n/1
    return *this + f; // richiama l'operatore + originario
}
};
int main(void) {
    frazione fr1(1,4), fr2(0.75), fr3;

    fr3=fr1; // l'assegnazione è automaticamente definita con il costruttore per copia?

    cout <<"fr1= "<<fr1.getnum()<<"/"<<fr1.getden();
    cout <<"\tfr2= "<<fr2.getnum()<<"/"<<fr2.getden();
    cout <<"\tfr3= "<<fr3.getnum()<<"/"<<fr3.getden()<<endl;
    fr3=fr1+fr2;
    cout << "Dopo la somma fr3=fr1+fr2 ..." <<endl;
    cout <<"fr1= "<<fr1.getnum()<<"/"<<fr1.getden();
    cout <<"\tfr2= "<<fr2.getnum()<<"/"<<fr2.getden();
    cout <<"\tfr3= "<<fr3.getnum()<<"/"<<fr3.getden()<< " = " << fr3.getreale()<<endl;
    fr3=fr1+5;
    cout << "Dopo la somma fr3=fr1+5 ..." <<endl;
    cout <<"fr1= "<<fr1.getnum()<<"/"<<fr1.getden();
    cout <<"\tfr2= "<<fr2.getnum()<<"/"<<fr2.getden();
    cout <<"\tfr3= "<<fr3.getnum()<<"/"<<fr3.getden()<< " = " <<fr3.getreale()<<endl;
    system("Pause");
}

```

```
    return 0;  
}
```



```
C:\Documents and Settings\HP\Documenti\Visual Studio 2010\Projects\oop7\Debug\oop7.exe  
fr1= 1/4      fr2= 3/4      fr3= 1/4  
Dopo la somma fr3=fr1+fr2 ...  
fr1= 1/4      fr2= 3/4      fr3= 1/1 = 1  
Dopo la somma fr3=fr1+5 ...  
fr1= 1/4      fr2= 3/4      fr3= 21/4 = 5.25  
Premere un tasto per continuare . . . _
```

5. Documentazione

6. Testing

OOP N° 8

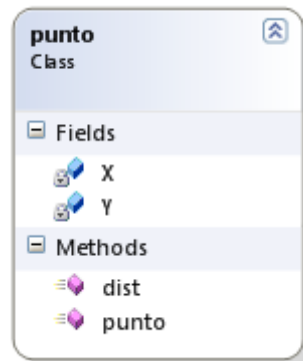
Definire la classe "punto" per rappresentare i punti del piano. Calcolare la distanza del punto dall'origine degli assi.

1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

3. Costruzione del modello



4. Codifica

Dev-C++ 4.9.9.2

```
#include <iostream>
```

```
#include <cmath>
```

```
using namespace std;
```

```
class punto {
```

```
    float X,Y;
```

```
    public:
```

```
    punto(float x=0, float y=0){ X=x; Y=y;}
```

```
    float dist();
```

```
// le funzioni amiche (friend) sono funzioni esterne alla classe che
```

```
// possono accedere a tutti gli attributi e metodi privati della classe
```

```
    friend ostream& operator<<(ostream&, punto&);
```

```
};
```

```
// overload dell'operatore << per l'oggetto cout della classe ostream
```

```
// serve a far funzionare cout anche con gli oggetti di classe punto
```

```
ostream& operator << (ostream &o, punto &p) {
```

```
    o<<'<<p.X<<','<<p.Y<<');
```

```
    return o;
```

```
}
```

```
int main(void)
```

```
{
```

```
    float x,y;
```

```
    cout << "inserisci X e Y " ; cin >> x >> y;
```

```
    punto p(x,y);
```

```
    cout << "Il punto inserito e' " <<p<< endl;
```

```
    cout << "la distanza dall'origine e' " << p.dist()<< endl;
```

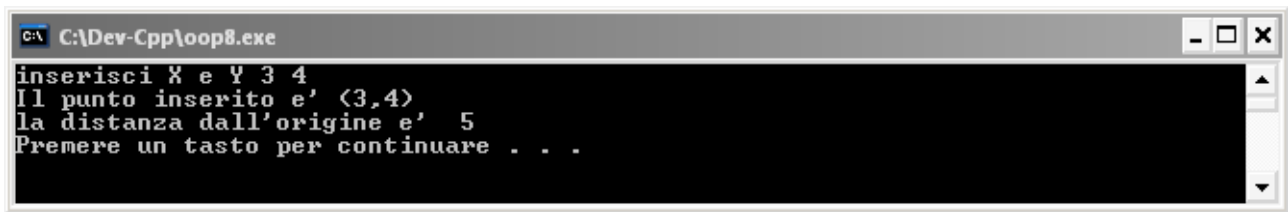
```
    system("Pause");
```

```
    return 0;
```

```
}
```


//implementazione di un metodo esterna alla classe di appartenenza

```
float punto::dist(){  
    return sqrt(X*X+Y*Y);  
}
```



Microsoft Visual C++

```
#include <iostream>
```

```
#include <cmath>
```

```
using namespace std;
```

```
class punto {
```

```
    float X,Y;
```

```
    public:
```

```
    punto(float x=0, float y=0){ X=x; Y=y;}
```

```
    float dist();
```

// le funzioni amiche (friend) sono funzioni esterne alla classe che possono accedere a tutti gli attributi e

//metodi privati della classe

```
    friend ostream& operator<<(ostream&, punto&);
```

```
};
```

// overload dell'operatore << per l'oggetto cout della classe ostream

// serve a far funzionare cout anche con gli oggetti di classe punto

```
ostream& operator<< (ostream &o, punto &p) {
```

```
    o<<'('<<p.X<<','<<p.Y<<')';
```

```
    return o; }
```

```
int main(void){
```

```
    float x,y;
```

```
    cout<< "inserisci X e Y " ; cin >> x >>y;
```

```
    punto p(x,y);
```

```
    cout<< "Il punto inserito e' " <<p<< endl;
```

```
    cout<< "la distanza dall'origine e' " << p.dist()<< endl;
```

```
    system("pause");
```

```
    return 0;}
```

//implementazione di un metodo esterna alla classe di appartenenza

```
float punto::dist(){
```

```
    return sqrt(X*X+Y*Y);}
```



5. Documentazione

6. Testing

OOP N°9

Progettare la Calcolatrice che possa svolgere le quattro operazioni elementari.

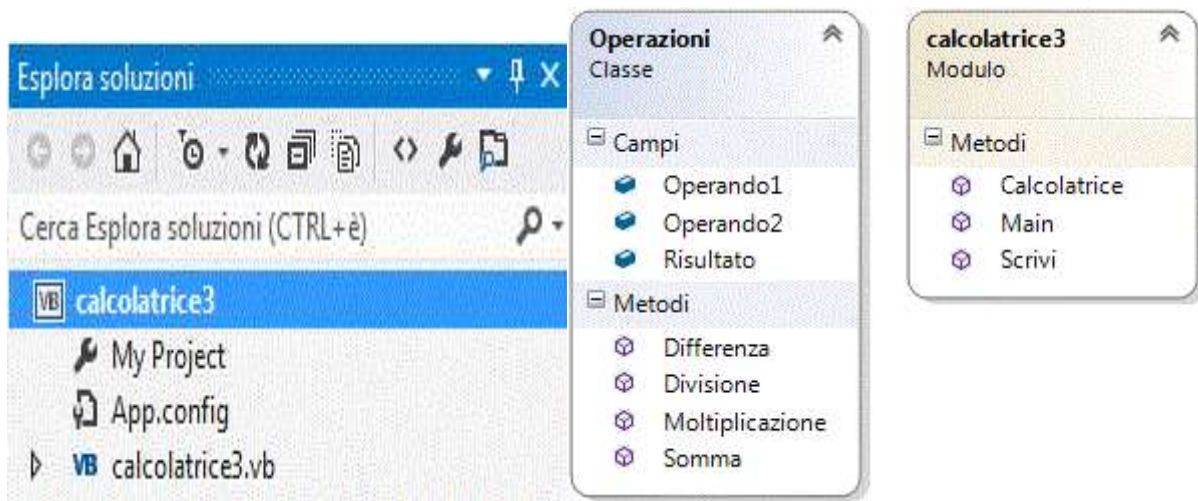
1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Gli unici attributi sono i due operandi Op1 e Op2. Definiamo quattro metodi, uno per ogni operazione elementare. Dal programma principale si propone dapprima l'inserimento del primo operando, poi del simbolo e infine del secondo operando, se il simbolo è +,-,*,/. L'acquisizione dei valori e degli operandi termina con il simbolo =.

3. Costruzione del modello



4. Codifica

Microsoft Visual C++

```
' classe
Public Class Operazioni
    ' attributi della classe
    Public Operando1 As Integer
    Public Operando2 As Integer
    Public Risultato As Integer
    ' metodi della classe
    Public Function Somma(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 + op2)
    End Function
    Public Function Differenza(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 - op2)
    End Function
    Public Function Moltiplicazione(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 * op2)
    End Function
    Public Function Divisione(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 \ op2)
    End Function
End Class
Module calcolatrice3
    Sub Main()
```

```

    Console.Clear()
    Calcolatrice()
    Console.ReadKey()
End Sub
Sub Calcolatrice()
    Dim Simbolo As String
    Dim op As Operazioni
    op = New Operazioni
    Console.Write("Inserisci il primo operando ")
    op.Operando1 = Console.ReadLine()
    Console.Write("Inserisci l'operatore: +, -, *, \: ")
    Simbolo = Console.ReadLine()
    While Simbolo <> "="
        Console.Write("Inserisci l'operando successivo ")
        op.Operando2 = Console.ReadLine()
        Select Case Simbolo
            Case "+"
                op.Risultato = op.Somma(op.Operando1, op.Operando2)
            Case "-"
                op.Risultato = op.Differenza(op.Operando1, op.Operando2)
            Case "*"
                op.Risultato = op.Moltiplicazione(op.Operando1, op.Operando2)
            Case "\"
                op.Risultato = op.Divisione(op.Operando1, op.Operando2)
        End Select
        op.Operando1 = op.Risultato
        Console.Write("Inserisci l'operatore, = per calcolare ")
        Simbolo = Console.ReadLine()
    End While
    Scrivi(op.Risultato)
End Sub
Sub Scrivi(ByVal r As Integer)
    Console.Write("Il risultato vale: {0}", r)
End Sub
End Module

```

```

file:///F:/Esercizi/Visual Studio 2012/Visual Basic/calcolatrice3/calcolatrice3/bin/Debug/calcolatrice...
Inserisci il primo operando 7
Inserisci l'operatore: +, -, *, \: -
Inserisci l'operando successivo 7
Inserisci l'operatore, = per calcolare =
Il risultato vale: 0

```

5. Documentazione

6. Testing

OOP N° 10

Progettare la Calcolatrice che possa svolgere le quattro operazioni elementari. e implementare l'ereditarietà per estensione.

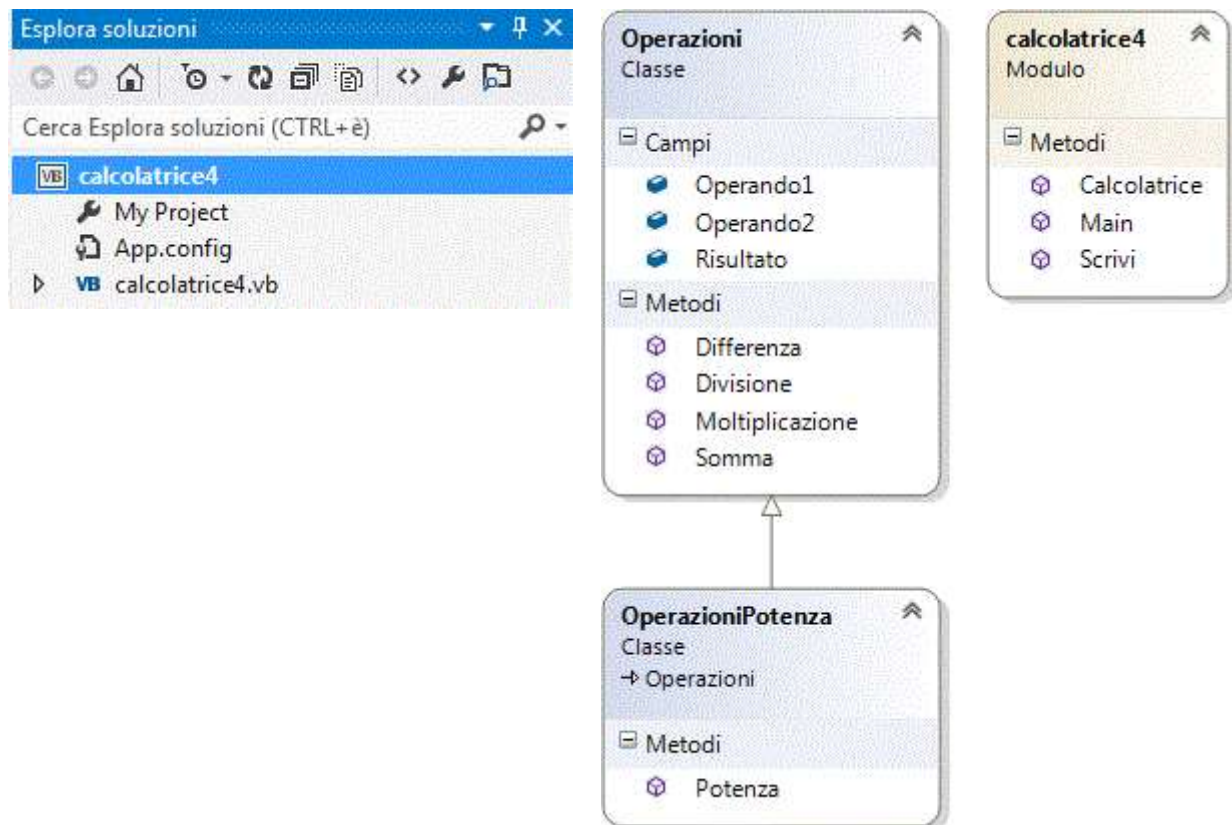
1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Si crea una nuova classe *OperazioniPotenza*, derivandola da un'esistente *Operazioni*, che eredita gli attributi e i metodi pubblici della classe *Operazioni* e al suo interno si codifica il nuovo metodo che permette l'elevamento a potenza.

3. Costruzione del modello



4. Codifica

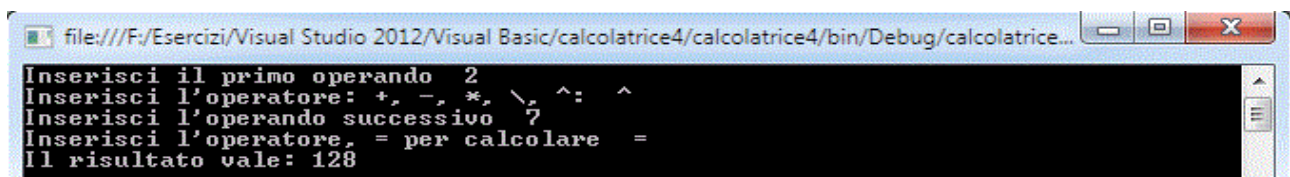
Microsoft Visual Basic

```
' classe
Public Class Operazioni
    ' attributi della classe
    Public Operando1 As Integer
    Public Operando2 As Integer
    Public Risultato As Integer
    ' metodi della classe
    Public Function Somma(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 + op2)
    End Function
    Public Function Differenza(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 - op2)
    End Function
```

```

    Public Function Moltiplicazione(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 * op2)
    End Function
    Public Function Divisione(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 \ op2)
    End Function
End Class
'sottoclasse
Public Class OperazioniPotenza
    Inherits Operazioni
    Public Function Potenza(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 ^ op2)
    End Function
End Class
Module calcolatrice4
    Sub Main()
        Console.Clear()
        Calcolatrice()
        Console.ReadKey()
    End Sub
    Sub Calcolatrice()
        Dim Simbolo As String
        Dim op As OperazioniPotenza
        op = New OperazioniPotenza
        Console.Write("Inserisci il primo operando ")
        op.Operando1 = Console.ReadLine()
        Console.Write("Inserisci l'operatore: +, -, *, \, ^: ")
        Simbolo = Console.ReadLine()
        While Simbolo <> "="
            Console.Write("Inserisci l'operando successivo ")
            op.Operando2 = Console.ReadLine()
            Select Case Simbolo
                Case "+"
                    op.Risultato = op.Somma(op.Operando1, op.Operando2)
                Case "-"
                    op.Risultato = op.Differenza(op.Operando1, op.Operando2)
                Case "*"
                    op.Risultato = op.Moltiplicazione(op.Operando1, op.Operando2)
                Case "\"
                    op.Risultato = op.Divisione(op.Operando1, op.Operando2)
                Case "^"
                    op.Risultato = op.Potenza(op.Operando1, op.Operando2)
            End Select
            op.Operando1 = op.Risultato
            Console.Write("Inserisci l'operatore, = per calcolare ")
            Simbolo = Console.ReadLine()
        End While
        Scrivi(op.Risultato)
    End Sub
    Sub Scrivi(ByVal r As Integer)
        Console.Write("Il risultato vale: {0}", r)
    End Sub
End Module

```



```
file:///F:/Esercizi/Visual Studio 2012/Visual Basic/calcolatrice4/calcolatrice4/bin/Debug/calcolatrice...
Inserisci il primo operando 2
Inserisci l'operatore: +, -, *, \, ^: ^
Inserisci l'operando successivo 7
Inserisci l'operatore, = per calcolare =
Il risultato vale: 128
```

5. Documentazione

6. Testing

OOP N° 11

Progettare la calcolatrice che esegue le quattro operazioni e implementare l'ereditarietà per overriding.

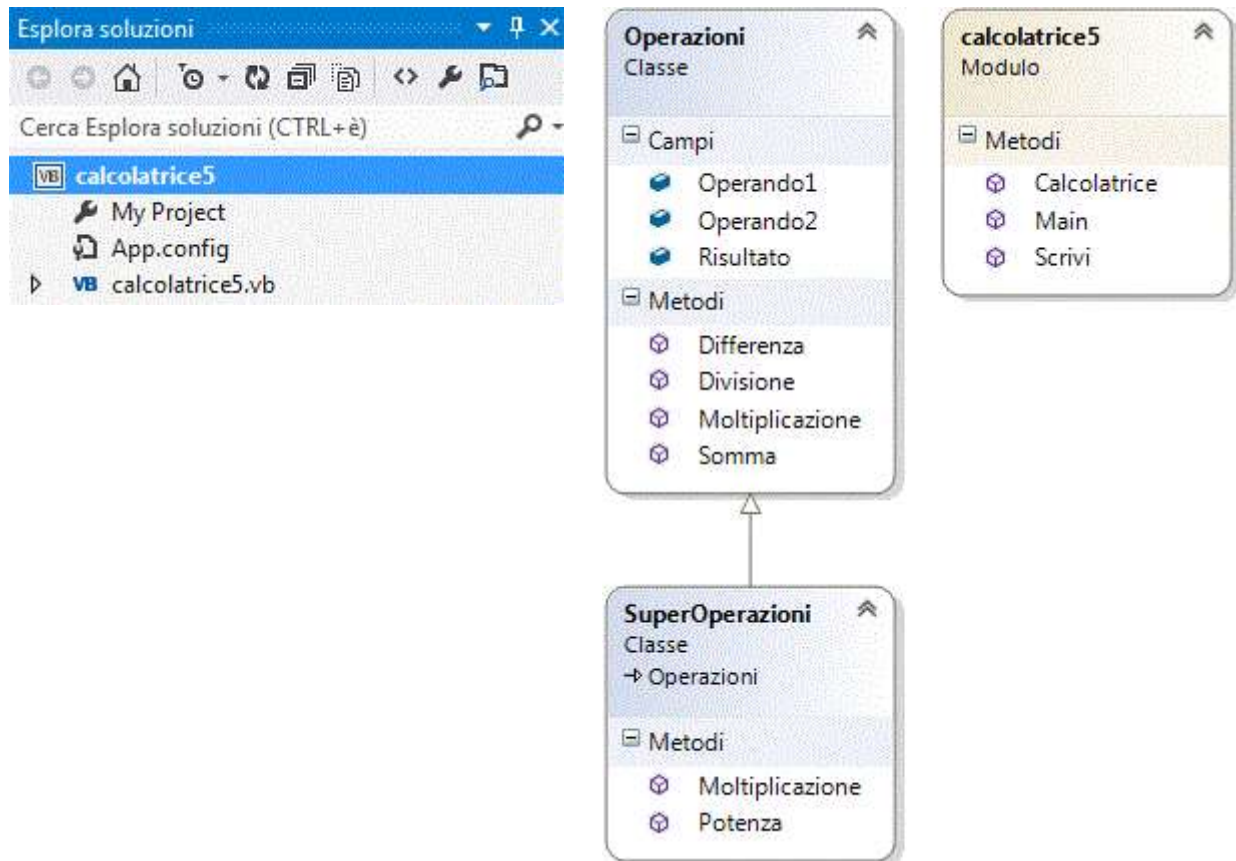
1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Si crea una nuova classe *SuperOperazioni*, derivandola da un'esistente *Operazioni*, che eredita gli attributi e i metodi pubblici della classe *Operazioni* e al suo interno si ridefinisce il metodo *Moltiplicazione* come serie di somme successive.

3. Costruzione del modello



4. Codifica

Microsoft Visual Basic

' superclasse

Public Class Operazioni

' attributi della classe

Public Operando1 As Integer

Public Operando2 As Integer

Public Risultato As Integer

' metodi della classe

Public Function Somma(ByVal op1 As Integer, ByVal op2 As Integer) As Integer

Return (op1 + op2)

End Function

Public Function Differenza(ByVal op1 As Integer, ByVal op2 As Integer) As Integer

Return (op1 - op2)

```

End Function
' la superclasse deve permettere la ridefinizione: Overridable
Public Overridable Function Moltiplicazione(ByVal op1 As Integer, ByVal op2 As
Integer) As Integer
    Return (op1 * op2)
End Function
Public Function Divisione(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
    Return (op1 \ op2)
End Function
End Class
' sottoclasse
Public Class SuperOperazioni
    Inherits Operazioni
    Public Function Potenza(ByVal op1 As Integer, ByVal op2 As Integer) As Integer
        Return (op1 ^ op2)
    End Function
    ' la sottoclasse ridefinisce i metodi: Overrides
    Public Overrides Function Moltiplicazione(ByVal op1 As Integer, ByVal op2 As Integer)
As Integer
        Dim p As Integer = 0
        If (op1 = 0) Then
            p = 0
        Else
            While (op1 <> 0)
                p = p + op2
                op1 = op1 - 1
            End While
        End If
        Return (p)
    End Function
End Class
Module calcolatrice5
    Sub Main()
        Console.Clear()
        Calcolatrice()
        Console.ReadKey()
    End Sub
    Sub Calcolatrice()
        Dim Simbolo As String
        Dim op As SuperOperazioni
        op = New SuperOperazioni
        Console.Write("Inserisci il primo operando ")
        op.Operando1 = Console.ReadLine()
        Console.Write("Inserisci l'operatore: +, -, *, \, ^: ")
        Simbolo = Console.ReadLine()
        While Simbolo <> "="
            Console.Write("Inserisci l'operando successivo ")
            op.Operando2 = Console.ReadLine()
            Select Case Simbolo
                Case "+"
                    op.Risultato = op.Somma(op.Operando1, op.Operando2)
                Case "-"
                    op.Risultato = op.Differenza(op.Operando1, op.Operando2)
                Case "*"

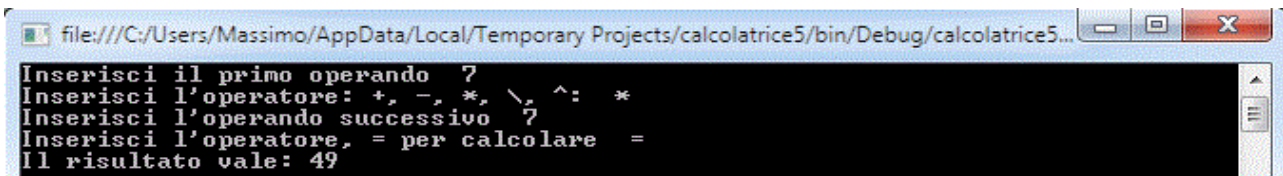
```



```

        op.Risultato = op.Moltiplicazione(op.Operando1, op.Operando2)
    Case "\"
        op.Risultato = op.Divisione(op.Operando1, op.Operando2)
    Case "^"
        op.Risultato = op.Potenza(op.Operando1, op.Operando2)
End Select
op.Operando1 = op.Risultato
Console.WriteLine("Inserisci l'operatore, = per calcolare ")
Simbolo = Console.ReadLine()
End While
Scrivi(op.Risultato)
End Sub
Sub Scrivi(ByVal r As Integer)
    Console.WriteLine("Il risultato vale: {0}", r)
End Sub
End Module

```



```

file:///C:/Users/Massimo/AppData/Local/Temporary Projects/calcolatrice5/bin/Debug/calcolatrice5...
Inserisci il primo operando 7
Inserisci l'operatore: +, -, *, \, ^: +
Inserisci l'operando successivo 49
Inserisci l'operatore, = per calcolare =
Il risultato vale: 49

```

5. Documentazione

6. Testing

OOP N° 12

Progettare la calcolatrice che esegue le quattro operazioni(polimorfismo).

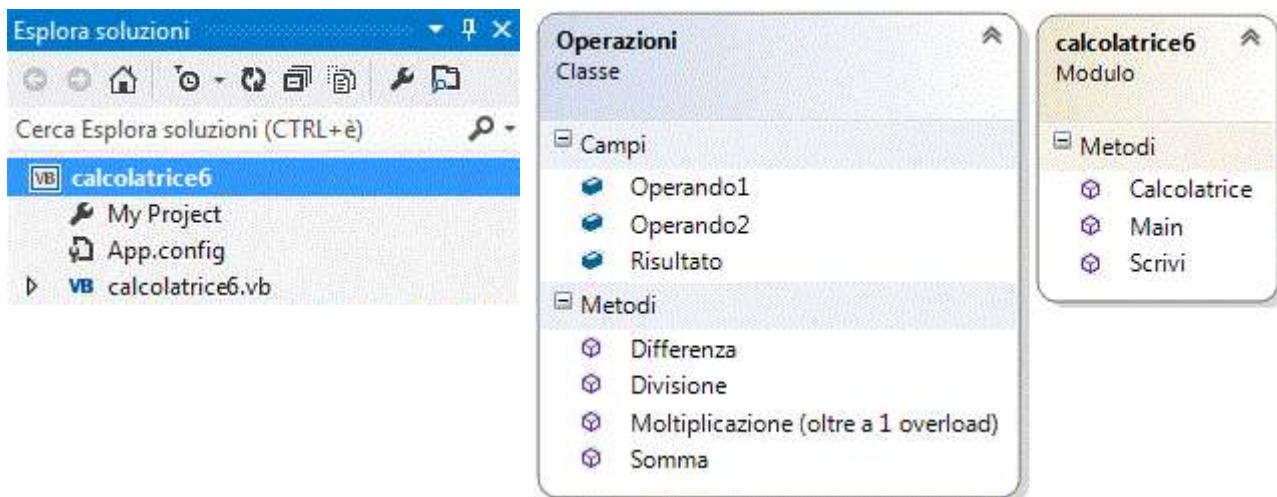
1. Identificazione del sistema

Il sistema è di tipo informatico.

2. Analisi dei dati

Si progettano due metodi *Moltiplicazione* con lo stesso nome che sono richiamati allo stesso modo ma con parametri di tipo diverso, un metodo lavora con numeri reali e l'altro con numeri interi.

3. Costruzione del modello



4. Codifica

Microsoft Visual Basic

' classe

Public Class Operazioni

' attributi della classe

Public Operando1 As Integer

Public Operando2 As Integer

Public Risultato As Integer

' metodi della classe

Public Function Somma(ByVal op1 As Integer, ByVal op2 As Integer) As Integer

Return (op1 + op2)

End Function

Public Function Differenza(ByVal op1 As Integer, ByVal op2 As Integer) As Integer

Return (op1 - op2)

End Function

Public Overloads Function Moltiplicazione(ByVal op1 As Integer, ByVal op2 As Integer)

As Integer

Return (op1 * op2)

End Function

Public Overloads Function Moltiplicazione(ByVal op1 As Double, ByVal op2 As Double)

As Double

Return (op1 * op2)

End Function

Public Function Divisione(ByVal op1 As Integer, ByVal op2 As Integer) As Integer

Return (op1 \ op2)

```

End Function
End Class
Module calcolatrice6
Sub Main()
    Console.Clear()
    Calcolatrice()
    Console.ReadKey()
End Sub
Sub Calcolatrice()
    Dim Simbolo As String
    Dim op As Operazioni
    op = New Operazioni
    Console.WriteLine("Inserisci il primo operando ")
    op.Operando1 = Console.ReadLine()
    Console.WriteLine("Inserisci l'operatore: +, -, *, \: ")
    Simbolo = Console.ReadLine()
    While Simbolo <> "="
        Console.WriteLine("Inserisci l'operando successivo ")
        op.Operando2 = Console.ReadLine()
        Select Case Simbolo
            Case "+"
                op.Risultato = op.Somma(op.Operando1, op.Operando2)
            Case "-"
                op.Risultato = op.Differenza(op.Operando1, op.Operando2)
            Case "*"
                op.Risultato = op.Moltiplicazione(op.Operando1, op.Operando2)
            Case "\"
                op.Risultato = op.Divisione(op.Operando1, op.Operando2)
        End Select
        op.Operando1 = op.Risultato
        Console.WriteLine("Inserisci l'operatore, = per calcolare ")
        Simbolo = Console.ReadLine()
    End While
    Scrivi(op.Risultato)
    Console.WriteLine()
End Sub
Sub Scrivi(ByVal r As Integer)
    Console.WriteLine("Il risultato vale: {0}", r)
End Sub
End Module

```

```

file:///F:/Esercizi/Visual Studio 2012/Visual Basic/calcolatrice6/calcolatrice6/bin/Debug/calcolatrice...
Inserisci il primo operando 2.2
Inserisci l'operatore: +, -, *, \: *
Inserisci l'operando successivo 2.2
Inserisci l'operatore, = per calcolare =
Il risultato vale: 484

```

5. Documentazione

6. Testing

PIRRÒ PAOLA

Dip. Informatica Industriale
I.T.I.S. "Giacomo Fauser"
Via Ricci, 14
28100 Novara Italy
tel. +39 0321482411
fax +39 0321482444
<http://www.fauser.edu>
pirro@fauser.edu