---

**a005prim.cpp**

```
#include <iostream.h>

main()
{
cout << "Questo e' il mio primo programma C++.\n";
return 0;
}
```

---

**a010prim.cpp**

```
#include<iostream.h>
main(){cout<<"Anche questo potrebbe essere il mio primo programma.\n";return 0;}
```

---

**a015prim.cpp**

```
             #include <iostream.h>

        main
        (
)
              {     cout       <<


              "... e anche questo!\n"

         ;                    return
           0
                                ;
                          }
```

---

**a020comm.cpp**

```
// Questo e' un <commento>
#include <iostream.h>
main// un commento puo' cominciare in qualsiasi posizione
    // ma termina alla fine della riga
( // la parentesi chiusa alla fine di questa riga fa parte del commento)
)

{  // comincia il corpo del programma
cout << "Commentiamo!\n";
return 0;}

// questo commento si trova dopo la fine del programma
```

---

**a025prim.cpp**

```
#include <iostream.h>  // Direttiva per il preprocessore: include il file
                       // di intestazione 'iostream.h' sul quale si basa
                       // il sistema di I/O del C++.
```

```
main()                  // Intestazione della funzione principale (main):
                        // indica dove ha inizio l'esecuzione del programma.
                        // La coppia di parentesi tonde e' obbligatoria e
                        // potra' in futuro contenere dei 'parametri'.

{                       // Parentesi graffa aperta: delimitatore di apertura
                        // del corpo della funzione main.

cout << "Primo esempio\n";  // Visualizza la frase 'Primo esempio' e
                        // posiziona il cursore all'inizio della riga
                        // successiva;
                        // cout sta per "console output" e rappresenta
                        // il flusso standard di output;
                        // il simbolo << e' l'operatore di inserimento

return 0;               // La funzione (main) restituisce il valore 0

}                       // Parentesi graffa chiusa: delimitatore di chiusura
                        // del corpo della funzione main
```

---

### a030altr.cpp

```
// nuovi concetti: sequenze di <<
#include <iostream.h>

main()
{
cout << "Altro " << "esempio\n";  // Visualizza 'Altro esempio'
        // "Altro " ed "esempio\n" sono stringhe letterali;
        // '\n' (newline: a capo) e' una 'sequenza di escape'
return 0;
}
```

---

### a035cout.cpp

```
// nuovi concetti: costante intera, espressione intera
// nuovi elementi: -, *, +
#include <iostream.h>

main() {
cout << 4 - 3 * 6 + 2 << "\n";   // visualizza il risultato dell'espressione:
                                 // -12
return 0;
}
```

---

### a040cout.cpp

```
// nuovi elementi: endl
#include <iostream.h>

main() {
cout << "4 - 3 * 6 + 2 = " << 4 - 3 * 6 + 2 << endl;
    // 'endl' e' un "manipolatore di stream" che
    // introduce un carattere di fine riga
return 0;
}
```

```
/* Output:
4 - 3 * 6 + 2 = -12
*/
```

---

### a045intv.cpp

```
// nuovi concetti: variabile, assegnamento
// nuovi elementi: int, =
#include <iostream.h>

main() {
int a;             // dichiara 'a' come variabile di tipo 'int':
                   // a puo' assumere un qualsiasi valore intero
                   // nell'intervallo -32768 ... +32767
a = -2;            // assegna il valore -2 alla variabile a
cout << a << endl; // visualizza il contenuto della variabile a
return 0;
}
```

---

### a050iniz.cpp

```
// nuovi concetti: dichiarazione e inizializzazione di una variabile
//                 espressione con variabili
#include <iostream.h>

main() {

int x = 12;       // dichiara e inizializza la variabile x

cout << x*x+2*x;  // visualizza il risultato dell'espressione: '168'
cout << endl;
return 0;
}
```

---

### a055asse.cpp

```
// nuovi concetti: dichiarazione di piu' variabili
//                 dichiarazione e inizializzazione nel corpo del programma
#include <iostream.h>

main()
{
int x, y1;       // dichiara le variabili x e y1
x = 77;
y1 = 88;
int y2 = 55;     // dichiara e inizializza a 55 la variabile y2
cout << x << ", " << y1 << ", " << y2 << endl;
return 0;
}

/* Output:
77, 88, 55
*/
```

---

### a060iniz.cpp

```
// nuovi concetti: dichiarazione di piu' variabili
```

```
//                  con inizializzazione di alcune
#include <iostream.h>

main()
{
int n1, n2 = 55, n3, n4, n5 = 44, n6;
cout << n2 << ", " << n5 << endl;
return 0;
}

/* Output:
55, 44
*/
```

---

### a065asse.cpp

```
// nuovi concetti: assegnazione come espressione
#include <iostream.h>

main()
{
int m, n;
m = (n = 66) + 9;     // (n = 66) e' un'espressione di assegnazione
cout << m << ", " << n << endl;
return 0;
}

/* Output:
75, 66
*/
```

---

### a070cin.cpp

```
// nuovi concetti: input da tastiera
// nuovi elementi: cin, >>
#include <iostream.h>

main() {
int x;

cout << "Introduci un numero intero: ";
cin >> x;       // assegna ad x il numero intero
                // introdotto da tastiera
                // cin sta per "console input" e rappresenta
                // il flusso standard di input;
                // il simbolo >> e' l'operatore di input

cout << "Il quadrato di " << x << " e' " << x*x << endl;
    // visualizza il quadrato del numero introdotto da tastiera
return 0;
}
```

---

### a075espr.cpp

```
// nuovi concetti: divisione tra interi
// nuovi elementi: /
#include <iostream.h>

main() {
cout << "2 + 5 * 7 / 2 = "
```

```
     << 2 + 5 * 7 / 2               // visualizza il risultato (19)
     << endl;                       // e poi va a capo
cout << "(2 + 5) * 7 / 2 = "
     << (2 + 5) * 7 / 2             // risultato: 24
     << endl;
return 0;
}

/* Output:
2 + 5 * 7 / 2 = 19
(2 + 5) * 7 / 2 = 24
*/
```

---

### a080oper.cpp

```
// nuovi elementi: %
#include <iostream.h>

main()
{
int m = -14, n = 5, q = m/n, r = m%n;
cout << "m = " << m << endl;
cout << "n = " << n << endl;
cout << "q = " << q << endl;
cout << "r = " << r << endl;
cout << "q*n + r = " << "(" << q << ")*(" << n << ") + "
     << r << " = " << q*n + r << " = " << m << endl;
return 0;
}

/* Output:
m = -14
n = 5
q = -2
r = -4
q*n + r = (-2)*(5) + -4 = -14 = -14
*/
```

---

### a085mani.cpp

```
// nuovi concetti: manipolatori di stream
// nuovi elementi: hex, dec, oct, \t
#include <iostream.h>

main() {
int n = 510;
cout << "  \tEsadecimale:\t " << hex << n << "\t " << n
     << "\n\t   Decimale:\t " << dec << n << "\t " << n
     << "\n\t     Ottale:\t " << oct << n << "\t " << n << endl;
cout << "Introduci un intero in ottale: ";
cin >> oct >> n;
cout << "  \tEsadecimale:\t " << hex << n << "\t " << n
     << "\n\t   Decimale:\t " << dec << n << "\t " << n
     << "\n\t     Ottale:\t " << oct << n << "\t " << n << endl;
cout << "Introduci un intero in esadecimale: ";
cin >> hex >> n;
cout << "  \tEsadecimale:\t " << hex << n << "\t " << n
     << "\n\t   Decimale:\t " << dec << n << "\t " << n
     << "\n\t     Ottale:\t " << oct << n << "\t " << n << endl;
return 0;
}
```

---

**a090if.cpp**

```
// nuovi concetti: selezione, espressione di relazione
// nuovi elementi: if, <
#include <iostream.h>

main() {
int n;
cout << "Introduci un numero intero: ";
cin >> n;
if (n < 0) n = -n;   // se n<0 allora cambia segno
cout << "valore assoluto = " << n << endl;
return 0;
}
```

---

**a095else.cpp**

```
// nuovi concetti: selezione a due vie
// nuovi elementi: ==, else
#include <iostream.h>

main() {
int n;
cout << "Introduci un numero intero: ";
cin >> n;
cout << "Il numero " << n << " e' ";
if (n % 2 == 0) cout << "pari";   // se n e' divisibile per 2
                                  // visualizza 'pari'
else cout << "dispari";           // altrimenti visualizza 'dispari'
cout << endl;
return 0;
}
```

---

**a100else.cpp**

```
// nuovi concetti: diverso da zero equivale a vero
#include <iostream.h>

main() {
int n;
cout << "Introduci un numero intero: ";
cin >> n;
cout << "Il numero " << n << " e' ";
if (n % 2) cout << "dispari";
else cout << "pari";
cout << endl;
return 0;
}
```

---

**a105cort.cpp**

```
// nuovi concetti: operatore AND
//                 cortocircuito nel calcolo di una espressione logica
// nuovi elementi: >, &&
#include <iostream.h>

main()
{
```

```
int n, d;
cout << "Introduci due interi positivi: ";
cin >> n >> d;
if (d > 0 && n%d == 0) cout << d << " e' un divisore di " << n << endl;
else cout << d << " non e' un divisore di " << n << endl;
return 0;
}
```

---

### a110whil.cpp

```
// nuovi concetti: iterazione
// nuovi elementi: while
#include <iostream.h>

// Calcola il Massimo Comune Divisore (MCD) di due numeri interi
// applicando l'algoritmo di Euclide

main() {
int numeroA, numeroB, a, b, r;   // dichiara 5 variabili di tipo int
cout << "----- Massimo Comune Divisore -----\n";
cout << "Introdurre due numeri interi positivi:\n";
cin >> numeroA >> numeroB;  // legge due interi da tastiera e li assegna
                            // rispettivamente a numeroA e numeroB
a = numeroA;    // assegna ad a il valore di numeroA
b = numeroB;    // assegna a  b il valore di numeroB
r = a % b;      // assegna a r il resto di a diviso b

while (r > 0)   // mentre r e' maggiore di 0 continua ad eseguire
                // le seguenti istruzioni (racchiuse tra '{' e '}')
    {
    a = b;
    b = r;
    r = a % b;
    }

// il ciclo 'while' e' terminato: r non e' piu' maggiore di 0
// (cioe', e' uguale a 0) => b e' il MCD

cout << "MCD (" << numeroA << "," << numeroB << ") = " << b << endl;
return 0;
}
```

---

### a115dowh.cpp

```
// nuovi elementi: do-while
#include <iostream.h>

// Calcola il Massimo Comune Divisore (MCD) di due numeri interi

main() {
int a, b, r;

cout << "----- Massimo Comune Divisore -----\n";
cout << "Introdurre due numeri interi positivi:\n";
cin >> a >> b;
cout << "MCD (" << a << "," << b << ") = ";

do               // esegui le istruzioni racchiuse tra '{' e '}'
   {
   r = a % b;    // ...
   a = b;
   b = r;        // ...
   }
```

```
while (r > 0);   // mentre r e' maggiore di 0

// il ciclo 'do-while' e' terminato: r e' uguale a 0  => a e' il MCD

cout << a << endl;
return 0;
}
```

---

### a120mcd.cpp

```
// nuovi concetti: cicli annidati
#include <iostream.h>

// Calcola il Massimo Comune Divisore (MCD) di due numeri interi

main() {
int a, b, r;
cout << "----- Massimo Comune Divisore -----\n";
cout << "Introdurre due numeri interi positivi (0 0 per finire):\n";
cin >> a >> b;

while (a > 0 && b > 0)   // il ciclo termina quando il prodotto da' 0

   {     // inizio blocco del ciclo esterno
   cout << "MCD (" << a << "," << b << ") = ";

   do
      {            // inizio blocco del ciclo interno
      r = a % b;
      a = b;
      b = r;
      }            // fine blocco del ciclo interno
      while (r);


   cout << a << endl;
   cin >> a >> b;
   }     // fine blocco del ciclo esterno

return 0;
}
```

---

### a125incr.cpp

```
// nuovo concetti: preincremento e postincremento
// nuovi elementi: ++
#include <iostream.h>

main()
{
int m = 66, n;
n = ++m;
cout << "m = " << m << ", n = " << n << endl;
n = m++;
cout << "m = " << m << ", n = " << n << endl;
cout << "m = " << m++ << endl;
cout << "m = " << m << endl;
cout << "m = " << ++m << endl;
return 0;
}

/* Output:
m = 67, n = 67
```

```
m = 68, n = 67
m = 68
m = 69
m = 70
*/
```

---

### a130blok.cpp

```
// nuovi concetti: variabile locale al blocco
#include <iostream.h>

main()
{
int x, y;
cout << "Introduci due interi: ";
cin >> x >> y;
if (x > y)
   {
   int temp = x;    // la variabile temp e' locale al blocco
   x = y;           // racchiuso tra le parentesi graffe
   y = temp;
   }
cout << "I due interi ordinati sono: ";
cout << x << " " << y << endl;
// cout << temp;  // questa istruzione produrrebbe un errore di compilazione
return 0;
}
```

---

### a135over.cpp

```
// nuovi concetti: overflow (traboccamento)
#include <iostream.h>

main()
{
int n = 32766;
cout << n++ << endl;
cout << n++ << endl;
cout << n++ << endl;
cout << n++ << endl;
return 0;
}

/* Output:
32766
32767
-32768
-32767
*/
```

---

### a140char.cpp

```
// nuovi concetti: costante carattere, caratteri come interi
// nuovi elementi: char
#include <iostream.h>

main()
{
char c = 'A';
cout << c++ << " ";
```

```
cout << c++ << " ";
cout << c++ << " ";
cout << c++ << endl;
c = 97;              // 97 e' il codice ASCII della lettera 'a'
cout << c++ << " ";
cout << c++ << " ";
cout << c++ << " ";
cout << c++ << endl;
return 0;
}

/* Output:
A B C D
a b c d
*/
```

---

### a145char.cpp

```
// nuovi concetti: type cast
#include <iostream.h>

main()
{
char c = 'A';
cout << c << " " << int(c) << endl;
c++;
cout << c << " " << int(c) << endl;
c++;
cout << c << " " << int(c) << endl;
return 0;
}

/* Output:
A 65
B 66
C 67
*/
```

---

### a150char.cpp

```
// nuovi concetti: input di caratteri
#include <iostream.h>

main()
{
char first, last;
cout << "Introduci le iniziali del tuo nome e cognome: ";
cin >> first >> last;
cout << "Salve, " << first << ". " << last << ".!\n";
return 0;
}
```

---

### a155limi.cpp

```
// nuovi elementi: limits.h
// nuovi concetti: short, long, signed, unsigned
#include <iostream.h>
#include <limits.h>
```

```
main()
{
cout << "minimum char = " << CHAR_MIN << endl;
cout << "maximum char = " << CHAR_MAX << endl;
cout << "minimum short = " << SHRT_MIN << endl;
cout << "maximum short = " << SHRT_MAX << endl;
cout << "minimum int = " << INT_MIN << endl;
cout << "maximum int = " << INT_MAX << endl;
cout << "minimum long = " << LONG_MIN << endl;
cout << "maximum long = " << LONG_MAX << endl;
cout << "minimum signed char = " << SCHAR_MIN << endl;
cout << "maximum signed char = " << SCHAR_MAX << endl;
cout << "maximum unsigned char = " << UCHAR_MAX << endl;
cout << "maximum unsigned short = " << USHRT_MAX << endl;
cout << "maximum unsigned int = " << UINT_MAX << endl;
cout << "maximum unsigned long = " << ULONG_MAX << endl;
return 0;
}
```

---

### a160for.cpp

```
// nuovi elementi: for, <=
#include <iostream.h>

main() {
int n;

// for (<inizializzazione>; <condizione di iterazione>; <aggiornamento>) <istruzione>
for (n = 1; n <= 10; n++)
   cout << n << " ";
cout << endl;
return 0;
}

/* Output:
1 2 3 4 5 6 7 8 9 10
*/
```

---

### a165asci.cpp

```
// nuovi concetti: tabella ASCII
#include <iostream.h>

main()
{
int ascii_val;

for (ascii_val = 32; ascii_val < 256; ascii_val++)
   {
   if (ascii_val % 4 == 0) cout << '\n';
   cout << '\t' << ascii_val << '\t' << char(ascii_val);
   }
cout << endl;

return 0;
}
```

---

### a170tabe.cpp

```
// nuovi concetti: variabile dichiarata nell'inizializzazione del for
#include <iostream.h>

main() {
int n;
cout << "Introduci un numero intero: ";
cin >> n;
cout << "Tabellina del " << n << endl;
for (int m = 1; m <= 10; m++)
        cout << n * m << endl;
return 0;
}
```

---

**a175for.cpp**

```
// nuovi elementi: operatore ","
#include <iostream.h>

main()
{
int m, n;
for (m = 1, n = 8; m < n; m++, n--)
   cout << "m = " << m << ", n = " << n << endl;
return 0;
}

/* Output:
m = 1, n = 8
m = 2, n = 7
m = 3, n = 6
m = 4, n = 5
*/
```

---

**a180for.cpp**

```
// nuovi concetti: for ( ; ; )
// nuovi elementi: +=, float, break
#include <iostream.h>
main()
{
int n, count = 0, sum = 0;
cout << "Introduci interi positivi.  Termina l'input con 0:\n";
for ( ; ; ) {
   cout << '\t' << count + 1 << ": ";
   cin >> n;
   if (n == 0) break;
   ++count;
   sum += n;
}
cout << "La media dei " << count << " numeri e' " << float(sum)/count << endl;
return 0;
}
```

---

**a185for.cpp**

```
// nuovo concetti: corpo del for vuoto
#include <iostream.h>
```

```
main()
{
int num, totale;

for (num = 1, totale = 0; num <= 10; totale += num++);

cout << "\nLa somma dei numeri da 1 a 10 e' "<< totale << endl;

return 0;
}
```

---

### a190divi.cpp

```
#include <iostream.h>

main() {
cout << "Calcola i divisori di un numero intero." << endl;
int numero;
do {
   cout << "Introduci un numero intero (<=0 per finire): " << endl;
   cin >> numero;
   if (numero > 0) {
      cout << "i divisori di " << numero << " sono: " << endl;
      for (int divisore = 2; divisore <= numero; divisore++)
         if (numero % divisore == 0)
            cout << divisore << endl;
      }
   cout << endl;
   } while (numero > 0);
return 0;
}
```

---

### a195ando.cpp

```
// nuovi elementi: >=, ||
#include <iostream.h>

main() {
int n;
do {
   cout << "Dammi un numero tra 10 e 20." << endl;
   cin >> n;
   if (n >= 10  &&  n <= 20) cout << "Grazie!";
   else cout << "Il tuo numero non e' compreso nell'intervallo 10-20\n";
   } while (n < 10  ||  n > 20);
return 0;
}
```

---

### a200depo.cpp

```
// nuovo concetti: variabili di tipo float
#include <iostream.h>

main() {
int anni, contanni;
float npercento, euro, interesse_corrente_anno, interesse_totale;

cout << "                      Calcolo dell'interesse accumulato\n";
```

```
cout << "                     ---------------------------------\n";
cout << "Introdurre la durata del deposito in anni (max 30), la cifra "
     << "depositata\n";
cout << "in euro e il tasso di interesse\n";
cin >> anni >> euro >> npercento;
cout << endl << endl;
if (anni<=0 || anni>30 || euro<=0 || npercento<=0) {
   cout << " dati forniti:  anni == " << anni
        << ",  euro == " << euro
        << ",  npercento == " << npercento
        << "\n dati non validi  -  programma interrotto.\n";
        }
else {
   cout << "   introiti su un deposito di                  " << euro
        << " euro\n"
        << "   tenuto per                                  " << anni
        << " anni\n"
        << "   ad un interesse annuale composto del        " << npercento
        << " percento\n"
        << " ------------------------------------------------------------------\n"
        << "      numero       interesse       capitale      interesse\n"
        << "       anno         annuale                     accumulato\n"
        << " ------------------------------------------------------------------\n";
   for (contanni=1, interesse_totale=0; contanni<=anni; contanni++) {
      interesse_corrente_anno = (euro * npercento) / 100;
      interesse_totale += interesse_corrente_anno;
      euro += interesse_corrente_anno;
      cout << contanni << " " << interesse_corrente_anno << " " << euro
           << " " << interesse_totale << endl;
      }
   cout << " ------------------------------------------------------------------\n";
   }
return 0;
}
```

---

**a205depo.cpp**

```
// nuovi elementi: iomanip.h, setw, setprecision, setf, ::
#include <iomanip.h>  // permette di usare setw, setprecision e setf
                      // include automaticamente iostream.h

main() {
int anni, contanni;
float npercento, euro, interesse_corrente_anno, interesse_totale;

cout << "                     Calcolo dell'interesse accumulato\n";
cout << "                     ---------------------------------\n";
cout << "Introdurre la durata del deposito in anni (max 30), la cifra "
     << "depositata\n";
cout << "in euro e il tasso di interesse\n";
cin >> anni >> euro >> npercento;
cout << endl << endl;
if (anni<=0 || anni>30 || euro<=0 || npercento<=0) {
   cout << " dati forniti:  anni == " << anni
        << ",  euro == " << euro
        << ",  npercento == " << npercento
        << "\n dati non validi  -  programma interrotto.\n";
   }
else {
   cout << "   introiti su un deposito di                  "
        << setw(8) << euro
        << " euro\n"
        << "   tenuto per                                  "
        << setw(8) << anni
        << " anni\n"
        << "   ad un interesse annuale composto del        "
```

```
        << setw(8) << npercento
        << " percento\n"
        << " ------------------------------------------------------------------\n"
        << "      numero       interesse       capitale      interesse\n"
        << "       anno         annuale                     accumulato\n"
        << " ------------------------------------------------------------------\n";
   cout.setf(ios::showpoint);
   cout.setf(ios::fixed, ios::floatfield);
   for (contanni=1, interesse_totale=0; contanni<=anni; contanni++) {
      interesse_corrente_anno = (euro * npercento) / 100;
      interesse_totale += interesse_corrente_anno;
      euro += interesse_corrente_anno;
      cout << setprecision(3);
      cout << setw(10) << contanni;
      cout << setw(18) << interesse_corrente_anno;
      cout << setw(16) << euro;
      cout << setw(14) << interesse_totale << endl;
      }
   cout << " ------------------------------------------------------------------\n";
   }
return 0;
}
```

---

### a210cast.cpp

```
// nuovi concetti: assegnare un reale a un intero
#include <iostream.h>

main()
{
double v = 1234.56789;
int n = v;
cout << "v = " << v << ", n = " << n << endl;
n = int(v);
cout << "v = " << v << ", n = " << n << endl;
return 0;
}

/* Output:
v = 1234.57, n = 1234
v = 1234.57, n = 1234
*/
```

---

### a215scop.cpp

```
// nuovi concetti: ambito di visibilita' (scope)
#include <iostream.h>

main()
{                        // inizio ambito di main()
int x = 22;
   {                 // inizio ambito di blocco interno
   int x = 33;
   cout << "Nel blocco interno a main(): x = " << x << endl;
   }                 // fine ambito di blocco interno
cout << "In main(): x = " << x << endl;
return 0;
}                        // fine ambito di main()

/* Output:
Nel blocco interno a main(): x = 33
In main(): x = 22
```

```
*/
```

---

### a220sqrt.cpp

```
// nuovi elementi: math.h, sqrt
#include <iostream.h>
#include <math.h>      // definisce le principali funzioni matematiche
                       // tra le quali la 'radice quadrata' (sqrt)
main() {
for (int i=0; i < 10; i++)
        cout << '\t' << i << '\t' << sqrt(i) << endl;
return 0;
}

/* Output:
        0       0
        1       1
        2       1.41421
        3       1.73205
        4       2
        5       2.23607
*/
```

---

### a225rand.cpp

```
// nuovi elementi: stdlib.h, rand
#include <iostream.h>
#include <iomanip.h>
#include <stdlib.h>   // definisce rand

// Questo programma genera dei numeri a caso (random)

main()
{
cout << "\n";
for (int i = 1; i <= 100; i++) {
   cout << setw(7) << rand();
   if (i%10 == 0) cout << endl;
   }
return 0;
}
```

---

### a230rand.cpp

```
// nuovi elementi: RAND_MAX
#include <stdlib.h>
#include <iomanip.h>

// Questo programma cerca di valutare l'intervallo dei numeri random

main() {
int n, min, max;
min = max = n = rand();
for (long i = 2; i <= 100000; i++) {
   n = rand();
   if (n < min) min = n;
   if (n > max) max = n;
   }
```

```
cout << "\nminimo = " << min << "\nmassimo = " << max << endl;
cout << "\nRAND_MAX = " << RAND_MAX << endl;
return 0;
}
```

---

### a235rand.cpp

```
// nuovi elementi: randomize
#include <iostream.h>
#include <stdlib.h>

main()
{
randomize();       // inizializza il seme per il calcolo dei numeri random
for (int i = 1; i <= 10; i++) {
   cout << rand() << endl;
   }
cout << endl;
return 0;
}
```

---

### a240rand.cpp

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

// Questo programma genera numeri casuali (interi) nell'intervallo [min, max]

main()
{
randomize();
int min, max, quanti, i;
cout << "Introduci minimo e massimo: ";
cin >> min >> max;
int range = max - min + 1;
cout << "Quanti numeri vuoi generare? ";
cin >> quanti;
for (i = 0; i < quanti; i++) {
   int r = rand()%range + min;
   cout << r << " ";
   }
cout << endl;
return 0;
}
```

---

### a245sqr.cpp

```
// nuovi concetti: definizione di funzione
#include <iostream.h>

// La seguente funzione 'sqr' restituisce il quadrato del suo argomento
int sqr(int n) {
return n * n;
}

main() {
int i;
for (i=0; i < 6; i++)
```

```
   cout << '\t' << i << '\t' << sqr(i) << endl;
cout << endl;
return 0;
}

/* Output:
        0       0
        1       1
        2       4
        3       9
        4       16
        5       25
*/
```

---

### a250sqr.cpp

```
// nuovi concetti: prototipo di funzione
#include <iostream.h>

int sqr(int);           // prototipo della funzione 'sqr'

main() {
for (int i=0; i < 6; i++)
   cout << '\t' << i << '\t' << sqr(i) << endl;
cout << endl;
return 0;
}

int sqr(int n) {
return n * n;
}
```

---

### a255max.cpp

```
// nuovi concetti: definizione di funzione con piu' parametri e piu' return
#include <iostream.h>

int max(int, int);

main() {
int m, n;
do {
  cout << "Introduci due interi (0 0 per finire): ";
  cin >> m >> n;
  if (m || n)
     cout << "Il massimo tra " << m << " e " << n << " e' "
          << max(m,n) << endl;
  } while (m || n);
return 0;
}

int max(int x, int y)
{
if (x < y) return y;
else return x;
}
```

---

### a260mcd.cpp

```
// nuovi elementi: int main()
#include <iostream.h>

int mcd(int,int);

int main() {
int a, b;
cout << "----- Massimo Comune Divisore -----\n";
cout << "Introdurre due numeri interi positivi (0 0 per finire):\n";
cin >> a >> b;
while (a > 0 && b > 0) {
   cout << "MCD (" << a << "," << b << ") = " << mcd(a,b) << endl;
   cin >> a >> b;
   }
return 0;
}


int mcd(int a, int b) {
// Calcola il Massimo Comune Divisore (MCD) di a e b
int r;
do {
   r = a % b;
   a = b;
   b = r;
   } while (r > 0);
return a;
}
```

---

### a265este.cpp

```
// nuovi concetti: caratteri estesi
// nuovi elementi: conio.h, getch()
#include <iostream.h>
#include <conio.h>   // definisce, tra l'altro, getch()

main()
{
int c, esteso;

while ((c = getch()) != 27) {  // 27 == codice ASCII di Esc
   if (c)
      cout << "Carattere normale: " << c << endl;
   else {
      esteso = getch();
      cout << "Carattere esteso: " << c << "-" << esteso << endl;
      }
   }
return 0;
}
```

---

### a270char.cpp

```
// nuovi concetti: espressione come argomento
// nuovi elementi: void, ctype.h, isdigit(), islower(), ecc...
#include <iostream.h>
#include <ctype.h>   // definisce, tra l'altro, le funzioni is....
#include <conio.h>

void printCharCategory(char c, int noprint) {
cout << "Il carattere [";
if (noprint)
   cout << "---";
```

```
else
   cout << c;
cout << "] e' un carattere ";
if      (isdigit(c)) cout << "numerico.\n";
else if (islower(c)) cout << "lettera minuscola.\n";
else if (isupper(c)) cout << "lettera maiuscola.\n";
else if (isspace(c)) cout << "di spaziatura.\n";
else if (iscntrl(c)) cout << "di controllo.\n";
else if (ispunct(c)) cout << "d'interpunzione.\n";
else                 cout << "Errore.\n";
}

void main() {
for (int c = 0; c < 128; c++) {
   cout << c << ": ";
   printCharCategory(c,c==8||c==10||c==13||c==26);
   if (c%20 == 19)
      if (!getch()) getch();  // funziona anche con tasti speciali!
   }
}
```

---

### a275rife.cpp

```
// nuovi concetti: passaggio dei parametri per riferimento
#include <iostream.h>

void swap(float& x, float& y) {
float temp = x;
x = y;
y = temp;
}

void main() {
float a = 27, b = -5.041;
cout << a << " " << b << endl;
swap(a,b);
cout << a << " " << b << endl;
}

/* Output:
27 -5.041
-5.041 27
*/
```

---

### a280cond.cpp

```
// nuovi elementi: ?: (operatore di espressione condizionale
//                     o operatore ternario)
#include <iostream.h>

main() {
int n;
cout << "Introduci un numero intero: ";
cin >> n;
cout << "Il numero " << n << " e' "
     << (n % 2 ? "dispari" : "pari") << endl;
return 0;
}
```

---

**a285fraz.cpp**

```
// nuovi elementi: /=, clrscr(), toupper(), getche()
#include <iostream.h>
#include <conio.h>   // definisce, tra l'altro, clrscr() e getche()
#include <ctype.h>   // definisce, tra l'altro, toupper()

void leggi(int&, int&, int&, int&);
int min_com_mul(int, int);
int mas_com_div(int, int);
void semplifica(int&, int&);
void scrivi(int, int);

void main() {
int n1, d1, n2, d2, num, den ;
char c;
clrscr();
do {
   leggi(n1,d1,n2,d2);
   cout << endl;
   cout << n1 << "/" << d1 << " + " << n2 << "/" << d2 << " = ";
   den = min_com_mul(d1,d2);
   num = den / d1 * n1 + den / d2 * n2;
   semplifica(num,den);
   scrivi(num,den);
   cout << endl << endl;
   do {
      cout << "Ancora? (S/N)";
      c = toupper(getche());
      } while ((c != 'S') && (c != 'N'));
   cout << endl << endl;
   } while (c == 'S');
cout << "The End!" << endl;
}

void leggi(int& n1, int& d1, int& n2, int& d2) {
do {
   cout << "Introduci numeratore e denominatore della prima frazione" << endl;
   cout << "diversi da 0 e separati da uno spazio" << endl;
   cin >> n1 >> d1;
   } while (n1*d1 == 0);
do {
   cout << "Introduci numeratore e denominatore della seconda frazione" << endl;
   cout << "diversi da 0 e separati da uno spazio" << endl;
   cin >> n2 >> d2;
   } while (n2*d2 == 0);
}

int min_com_mul(int d1, int d2) {
int ma = d1, mb = d2;
while ( ma != mb )
   ma > mb  ?  (mb += d2)  :  (ma += d1);
return ma;
}

int mas_com_div(int nn, int dd) {
while ( dd ) {
   int resto = nn % dd;
   nn = dd;
   dd = resto;
   }
return nn;
}

void semplifica(int& num, int& den) {
int mcd = mas_com_div(num,den);
num /= mcd;
den /= mcd;
}
```

```
void scrivi(int num, int den) {
cout << num;
if ( den != 1 )
   cout << "/" << den;
}
```

---

## a290para.cpp

```
// nuovi concetti: confronto fra il passaggio dei parametri
//                 per valore e per riferimento
#include <iostream.h>

int n;

void P(int x) {
x += 1;
cout << n << " " << x << " ";
}

void Q(int& x) {
x += 1;
cout << n << " " << x << " ";
}

void main() {
n = 3;
P(n);
cout << n << endl;
n = 3;
Q(n);
cout << n << endl;
}

/* Output:
3 4 3
4 4 4
*/
```

---

## a295ambi.cpp

```
// nuovi concetti: ambito degli identificatori
#include <iostream.h>

int a, b;

void P() {
int a = b+1;
b = a+1;
}

void Q() {
int b = a-1;
a = b-1;
}

void main() {
b = a = 10;
P();
Q();
cout << "a = " << a << ", b = " << b << endl;
}
```

```
/* Output:
a = 8, b = 12
*/
```

---

### a300scop.cpp

```
// nuovi concetti: visibilita'
#include <iostream.h>

int x;

void P1() {
cout << x << endl;
}

void P2() {
int x = 23;
x++;
P1();
}

void main() {
x = 12;
P2();
}

/* Output:
12
*/
```

---

### a305visi.cpp

```
// nuovi concetti: variabile globale, ambiti di visibilita'
#include <iostream.h>

void f();
void g();
int x = 11;

void main()
{
int x = 22;
   {
   int x = 33;
   cout << "Nel blocco interno a main(): x = " << x << endl;
   }
cout << "In main(): x = " << x << endl;
f();
g();
}

void f() {
int x = 44;
cout << "In f(): x = " << x << endl;
}

void g() {
cout << "In g(): x = " << x << endl;
}

/* Output:
Nel blocco interno a main(): x = 33
```

```
In main(): x = 22
In f(): x = 44
In g(): x = 11
*/
```

---

### a310side.cpp

```
// nuovi concetti: effetti collaterali
// nuovi elementi: *=
#include <iostream.h>

int f(int& a) {
a *= 2;
return a;
}

void main() {
int b = 1;
cout << 2*f(b) << " ";
b = 1;
cout << f(b) + f(b) << endl;
}

/* Output:
4 6
*/
```

---

### a315cin.cpp

```
// nuovi concetti: risultato di cin, ^Z
#include <iostream.h>

void main() {
int n, sum=0;
while (cin >> n)
   sum += n;
cout << "somma = " << sum << endl;
}
```

---

### a320prim.cpp

```
// nuovi concetti: stringa vuota ("")
// nuovi elementi: const
//*********************************************************************
//*                                                                   *
//*    Controlla se "n" e' un numero primo (2<=n<=nmax).              *
//*                                                                   *
//*********************************************************************
#include <iostream.h>
#include <math.h>
#include <conio.h>

const NMAX = 10000;    // equivale a  const int NMAX = 10000; !
const TRUE = 1;
const FALSE = 0;

int datobuono (int& n, int massimo) {
   //*****************************************************************
```

```
   //*                                                                *
   //*    Ottiene il successivo dato valido, se esiste, saltando     *
   //*    eventuali valori che siano al di fuori dell'intervallo     *
   //*    permesso.  Il valore trovato, se c'e', ritorna attraverso  *
   //*    il parametro riferimento "n".  Se si preme ^Z, o si        *
   //*    introduce un carattere non accettabile come intero, la     *
   //*    funzione restituisce "falso".  Il parametro valore         *
   //*    "massimo" determina l'intervallo accettabile dei dati.     *
   //*                                                                *
   //******************************************************************
int trovato = FALSE;    // abbiamo trovato un "n" valido?
while ( (!trovato) && (cin >> n) ) {
   if ( (n>=2) && (n<=massimo) ) trovato = TRUE;
   else cout << n << " fuori limiti: ignorato\n";
   }
return trovato;
}  // datobuono

int decidi (int n) {
   //*****************************************************************
   //*                                                               *
   //*   Determina se "n" e' un numero primo.                        *
   //*   Questo sottoprogramma ha un parametro valore; usa due       *
   //*   variabili locali, ma non fa riferimento a nessuna           *
   //*   variabile globale.                                          *
   //*                                                               *
   //*****************************************************************
int provadivisore,         // possibili divisori di "n"
    limite= int(sqrt(n));  // il piu' grande divisore possibile di "n"
for (provadivisore = 2; (provadivisore<=limite) && (n % provadivisore);
     provadivisore++);
return (provadivisore > limite);
}  // decidi

void mostra (int numero, int estprimo) {
   //****************************************************************
   //*                                                              *
   //*   Mostra la decisione del programma ("estprimo") sul         *
   //*   numero corrente.  Questo sottoprogramma ha due parametri   *
   //*   valore;  non usa variabili locali ne' fa riferimento       *
   //*   a variabili globali                                        *
   //*                                                              *
   //****************************************************************
cout << numero << (estprimo ? "" : " non")
     << " e' un numero primo" << endl;
}  // mostra

void main() {    // programma principale
int n;           // il valore da controllare (dato)
clrscr();
cout << "Controlla se numeri positivi non superiori a " << NMAX
     << " sono primi (^Z per finire):\n\n";
while (datobuono(n,NMAX))
   mostra(n,decidi(n));
}
```

---

## a325prim.cpp

```
// nuovi concetti: funzione con diversi punti di uscita
#include <iostream.h>
#include <math.h>

int NumeroPrimo(int n) {
float sqrtp = sqrt(n);
if (n < 2) return 0;
if (n == 2) return 1;
if (n % 2 == 0) return 0;
```

```
for (int d = 3; d <= sqrtp; d += 2)
  if (n % d == 0) return 0;
return 1;
}

void main() {
for (int n = 1; n < 50; n++)
   if (NumeroPrimo(n)) cout << n << " ";
cout << endl;
}
```

---

**a330leap.cpp**

```
// nuovi concetti: funzione con una sola istruzione (return)
//                 precedenza degli operatori
#include <iostream.h>

int bisestile(int y) {
return y % 4 == 0 && (y % 100 != 0 || y % 400 == 0);
}

void main() {
int anno;
do {
   cout << "Introdurre anno (0 per finire): ";
   cin >> anno;
   if (anno > 0)
      cout << anno << " "
           << (bisestile(anno) ? "" : "non ")
           << "e' un anno bisestile.\n";
   } while (anno > 0);
}
```

---

**a335retu.cpp**

```
// nuovi concetti: while (1) ...
// nuovi elementi: \a (segnale sonoro - beep)
#include <iostream.h>

int age(void) {
int n;
while (1) {
   cout << "Introdurre l'eta': ";
   cin >> n;
   if (n < 0) cout << "\a\tL'eta' non puo' essere negativa.";
   else if (n > 120) cout << "\a\tL'eta' non puo' essere oltre 120.";
   else return n;
   cout << "\n\tProva ancora.\n";
  }
}

void main() {
int a = age();
cout << "\nL'eta' e' " << a <<" anni.\n";
}
```

---

**a340aray.cpp**

```
// nuovi concetti: array
// nuovi elementi: [, ]
#include <iostream.h>

void main() {
int i, a[6];
cout << "Introduci 6 numeri interi.\n";
for (i = 0; i < 6; i++)
   cin >> a[i];
cout << endl;
for (i = 5; i >= 0; i--)
   cout << a[i] << endl;
}
```

---

**a345cost.cpp**

```
#include <iostream.h>

void main() {
const int n = 6;    // meglio usare costante!
int a[n], i;
cout << "Introdurre " << n << " interi\n";
for (i = 0; i < n; i++)
   cin >> a[i];
cout << endl;
for (i = n; i > 0; i--)
   cout << a[i-1] << endl;
}
```

---

**a350aray.cpp**

```
// nuovi concetti: inizializzazione di un array
#include <iostream.h>

void main() {
double a[4] = {22.2, 44.4, 66.6, 88.8};
for (int i = 0; i < 4; i++)
   cout << "a[" << i << "] = " << a[i] << endl;
}

/* Output:
a[0] = 22.2
a[1] = 44.4
a[2] = 66.6
a[3] = 88.8
*/
```

---

**a355csnt.cpp**

```
// nuovi concetti: parametri costanti
#include <iostream.h>

void f(int x, int& y, const int z) {
x += z;
y += z;
// z++;  // questa istruzione produrrebbe un errore di compilazione!
cout << "x = " << x << ", y = " << y << ", z = " << z << endl;
}
```

```
void main() {
int a = 22, b = 33, c = 44;
cout << "a = " << a << ", b = " << b << ", c = " << c << endl;
f(a,b,c);
cout << "a = " << a << ", b = " << b << ", c = " << c << endl;
}

/* Output:
a = 22, b = 33, c = 44
x = 66, y = 77, z = 44
a = 22, b = 77, c = 44
*/
```

---

### a360aray.cpp

```
// nuovi concetti: passaggio di array a funzioni
#include <iostream.h>

const int size = 100;

void getArray(double [], int&);

void printArray(const double [], const int);

void main() {
double a[size];
int n;
getArray(a,n);
cout << "L'array ha " << n << " elementi.\nQuesti sono:\n";
printArray(a,n);
}

void getArray(double a[], int& n) {
n = 0;
cout << "Introduci i dati. Termina con 0:\n";
for (n = 0; n < size; n++) {
  cout << n << ": ";
  cin >> a[n];
  if (a[n] == 0) break;
  }
}

void printArray(const double a[], const int n) {
for (int i = 0; i < n; i++)
  cout << '\t' << i << ": " << a[i] << endl;
}
```

---

### a365matr.cpp

```
// nuovi concetti: array bidimensionale, for annidato
#include <iostream.h>

void main() {
const int dim = 3;
int i, j, m[dim][dim];

cout << "Introdurre matrice (" << dim << "x" << dim << ")\n";
for (i=0; i<dim; i++)
   for (j=0; j<dim; j++)
      cin >> m[i][j];
cout << "\nMatrice introdotta:\n";
```

```
for (i=0; i<dim; i++) {
   for (j=0; j<dim; j++)
      cout << m[i][j] << " ";
   cout << endl;
   }
cout << "\nMatrice trasposta:\n";
for (i=0; i<dim; i++) {
   for (j=0; j<dim; j++)
      cout << m[j][i] << " ";
   cout << endl;
   }
}
```

---

### a370dadi.cpp

```
// Simulazione del lancio di due dadi
#include <iostream.h>
#include <stdlib.h>
#include <iomanip.h>

void main() {
unsigned long nlanci, i, max, a[11] = {0};
unsigned int j;
randomize();
cout << "Quanti lanci? ";
cin >> nlanci;
for (i=0; i<nlanci; i++)
   a[random(6) + random(6)]++;
for (max=a[0],i = 1; i < 11; i++)
   if (a[i]>max) max=a[i];

cout << "Risultato dei lanci:\n";

for (i = 0; i < 11; i++) {
   cout << setw(3) << i+2 << ":" << setw(9) << a[i] << "  ";
   for (j=1;j<=a[i]*60/max;j++) cout << '*';
   cout << endl;
   }
cout << "               ";
for (j=1;j<=6;j++) cout << "---------+";
cout << endl;
}
```

---

### a375stln.cpp

```
// nuovi elementi: string.h, strlen()
#include <iostream.h>
#include <string.h>

main()
{
cout << strlen("Hello, World.\n") << '\n';
cout << strlen("Hello, World.") << '\n';
cout << strlen("Hello, ") << '\n';
cout << strlen("H") << '\n';
cout << strlen("") << '\n';
return 0;
}

/* Output:
14
13
```

```
7
1
0
*/
```

---

### a380strn.cpp

```
// nuovi concetti: array di char
#include <iostream.h>

void main() {
char s[] = {'A', 'B', 'C', 'D'};
for (int i=0; i<sizeof(s); i++)
   cout << "s[" << i << "] = '" << s[i] << "', " << int(s[i]) << "\n";
}
```

---

### a385strn.cpp

```
// nuovi concetti: variabile stringa
#include <iostream.h>

void main() {
char s[] = "ABCD";     /* equivale a
char s[] = {'A', 'B', 'C', 'D', '\0'};  */
for (int i=0; i<sizeof(s); i++)
   cout << "s[" << i << "] = '" << s[i] << "', " << int(s[i]) << "\n";
}
```

---

### a390stio.cpp

```
// nuovi concetti: input-output di stringhe
#include <iostream.h>

void main() {
char word[80];
cout << "Introduci una sequenza di parole (^Z per terminare):\n";
while (cin >> word)
  cout << "\t\"" << word << "\"\n";
}
```

---

### a395stio.cpp

```
// nuovi concetti: ^Z -> stringa vuota
#include <iostream.h>

void main() {
char word[80];
cout << "Introduci una sequenza di parole (^Z per terminare):\n";
cin >> word;
while (word[0]) {
  cout << "\t\"" << word << "\"\n";
  cin >> word;
  }
}
```

## a400strl.cpp

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

void main() {
char s[] = "ABCDEFG";
cout << "strlen(" << s << ") = " << strlen(s) << endl;
cout << "strlen(\"\") = " << strlen("") << endl;
char buffer[80];
cout << "Introduci stringa: ";  cin >> buffer;
cout << "strlen(" << buffer << ") = " << strlen(buffer) << endl;
}
```

## a405getl.cpp

```
// nuovi elementi: cin.getline()
#include <iostream.h>

void main() {
char line[80];
do {
  cin.getline(line, 80);
  if (line[0]) cout << "\t[" << line << "]\n";
  } while (line[0]);
}
```

## a410maca.cpp

```
// nuovi concetti: array di stringhe
#include <iostream.h>

void main() {
char name[8][24];
int count;
cout << "Introduci non piu' di 8 nomi di non piu' di 23 caratteri:\n";
for (count=0; count < 8 && cin.getline(name[count],24); ++count)
        ;
cout << "I nomi sono:\n";
for (int i = 0; i < count; i++)
   cout << '\t' << i << ". [" << name[i] << "]" << endl;
}
```

## a415strp.cpp

```
// nuovi concetti: copiare una stringa su un'altra
// nuovi elementi: strcpy()
#include <iostream.h>
#include <string.h>

void main() {
char s1[] = "ABCDEFG";
char s2[] = "XYZ";
cout << "Prima di strcpy(s1,s2):\n";
cout << "\ts1 = [" << s1 << "], lunghezza = " << strlen(s1) << endl;
cout << "\ts2 = [" << s2 << "], lunghezza = " << strlen(s2) << endl;
```

```
strcpy(s1,s2);
cout << "Dopo strcpy(s1,s2):\n";
cout << "\ts1 = [" << s1 << "], lunghezza = " << strlen(s1) << endl;
cout << "\ts2 = [" << s2 << "], lunghezza = " << strlen(s2) << endl;
}
```

---

### a420stct.cpp

```
// nuovi concetti: concatenazione di stringhe
// nuovi elementi: strcat()
#include <iostream.h>
#include <string.h>

void main() {
char s1[11] = "ABCDEFG";
char s2[] = "XYZ";
cout << "Prima di strcat(s1,s2):\n";
cout << "\ts1 = [" << s1 << "], lunghezza = " << strlen(s1) << endl;
cout << "\ts2 = [" << s2 << "], lunghezza = " << strlen(s2) << endl;
strcat(s1,s2);
cout << "Dopo strcat(s1,s2):\n";
cout << "\ts1 = [" << s1 << "], lunghezza = " << strlen(s1) << endl;
cout << "\ts2 = [" << s2 << "], lunghezza = " << strlen(s2) << endl;
}
```

---

### a425sovr.cpp

```
// nuovi concetti: sovrapposizione di funzioni (function overloading)
#include <iostream.h>

int max(int, int);

int max(int, int, int);

double max(double, double);

void main() {
cout << max(99,77) << " " << max(55,66,33) << " " << max(3.4,7.2) << endl;
}

int max(int x, int y)
{
return (x > y ? x : y);
}

int max(int x, int y, int z)
{
int m = (x > y ? x : y);
return (z > m ? z : m);
}


double max(double x, double y)
{
return (x > y ? x : y);
}

/* Output:
99 66 7.2
*/
```

## a430exit.cpp

```
// nuovi elementi: exit()
#include <iostream.h>
#include <stdlib.h>  // definisce, tra l'altro, exit()

double inverso(double x)
{
if (x == 0) exit(1);
return 1.0/x;
}

void main() {
double x;
cout << "Introdurre un numero: ";
cin >> x;
cout << "1/" << x << " = " << inverso(x) << endl;
}
```

## a435dflt.cpp

```
// nuovi concetti: argomenti di default
#include <iostream.h>

double p(double, double, double =0, double =0, double =0);


void main() {
double x = 2.0003;
cout << "p(x,7) = " << p(x,7) << endl;
cout << "p(x,7,6) = " << p(x,7,6) << endl;
cout << "p(x,7,6,5) = " << p(x,7,6,5) << endl;
cout << "p(x,7,6,5,4) = " << p(x,7,6,5,4) << endl;
}

double p(double x, double a0, double a1, double a2, double a3)
{
return a0 + (a1 + (a2 + a3*x)*x)*x;
}

/* Output:
p(x,7) = 7
p(x,7,6) = 19.0018
p(x,7,6,5) = 39.0078
p(x,7,6,5,4) = 71.0222
*/
```

## a440cont.cpp

```
// nuovi concetti: ricorsione
#include <iostream.h>

void conta(int n) {
if ( n > 0 ) {
   cout << n << " ";
   conta(n-1);        // chiamata ricorsiva
   }
}

void main() {
```

```
conta(10);
cout << endl;
}

/* Output:
10 9 8 7 6 5 4 3 2 1
*/
```

---

### a445cont.cpp

```
#include <iostream.h>

void conta(int n) {
if ( n > 0 ) {
   conta(n-1);
   cout << n << " ";
   }
}

void main() {
conta(10);
cout << endl;
}

/* Output:
1 2 3 4 5 6 7 8 9 10
*/
```

---

### a450mcdr.cpp

```
#include <iostream.h>

int MCD(int m, int n) {
int r = m % n;
if (r == 0) return n;
else return MCD(n,r);
}

void main() {
int a, b;
cout << "----- Massimo Comune Divisore -----\n";
cout << "Introdurre due numeri interi positivi (0 0 per finire):\n";
cin >> a >> b;
while (a > 0 && b > 0)  {
   /* il ciclo termina quando almeno uno dei due numeri e' 0 */
   cout << "MCD (" << a << ","  << b << ") = " << MCD(a,b) << endl << endl;
   cin >> a >> b;
   }
}
```

---

### a455fact.cpp

```
/* prova di due sottoprogrammi per il calcolo del fattoriale:
   uno non ricorsivo, l'altro ricorsivo */
#include <iostream.h>

long double fattoriale1 (int n) {
/**********************************************/
```

```
/*                                              */
/*   Calcola n! per "n" intero positivo.        */
/*   Questo sottoprogramma e' non ricorsivo     */
/*                                              */
/************************************************/
long double partfact;
for (partfact = 1; n > 1; partfact *= n--)
   ;
return partfact;
}

long double fattoriale2 (int n) {
/************************************************/
/*                                              */
/*   Calcola n! per "n" intero positivo.        */
/*   Questo sottoprogramma e' ricorsivo         */
/*                                              */
/************************************************/
return n == 1 ? 1 : n*fattoriale2(n-1);
}

void main()  {
int n;
cout << "Calcolo del fattoriale\n";
do {
   cout << "Introdurre un numero intero positivo: ";
   cin >> n;
   if (n>0) {
      cout << n << "! calcolato in due modi:" << endl;
      cout << fattoriale1(n) << " - non ricorsivo" << endl;
      cout << fattoriale2(n) << " - ricorsivo" << endl;
      }
   } while (n>0);
}
```

---

### a460indi.cpp

```
#include <iostream.h>

void AllIndietro() {
int numero;
cout << "Dammi un numero (0 per finire): ";
cin >> numero;
if (numero) AllIndietro();
cout << numero << endl;
}

void main() {
AllIndietro();
}
```

---

### a465indi.cpp

```
#include <iostream.h>

int numero;

void AllIndietro() {
cout << "Dammi un numero (0 per finire): ";
cin >> numero;
if (numero) AllIndietro();
cout << numero << endl;
```

```
}

void main() {
AllIndietro();
}
```

---

### a470hano.cpp

```
/* Scrive la sequenza richiesta per risolvere il problema */
/* della "Torre di Hanoi"                                  */
#include <iostream.h>

unsigned long numeromosse;     /* numero di mosse  */

void muovidisco (int, int);
void muovitorre (int, int, int);


void main() {
int totale;           /* numero di dischi */

cout << "Torre di Hanoi\n";
cout << "Quanti dischi? ";
cin >> totale;
numeromosse = 0;
muovitorre (totale,1,3);
cout << "\nNumero totale mosse per " << totale
     << " dischi = " << numeromosse << endl;
}


void muovitorre (int altezza, int partenza, int arrivo) {
/* Muove una torre di altezza pari ai dischi dalla partenza */
/* all'arrivo                                               */
if (altezza > 0) {
   int ausiliario = 6-partenza-arrivo;
   muovitorre (altezza - 1,partenza,ausiliario);
   muovidisco (partenza,arrivo);
   muovitorre (altezza - 1,ausiliario,arrivo);
   }
}


void muovidisco (int sorgente, int destinazione) {
/* Scrive e muove da sorgente a destinazione */
numeromosse++;
cout << numeromosse << ": " << sorgente << " -> " << destinazione << endl;
}
```

---

### a475bino.cpp

```
/* Coefficienti binomiali o
   Triangolo di Pascal o
   Triangolo di Tartaglia */
#include <iostream.h>
#include <iomanip.h>

const NROWS = 10;

int binomial (int n, int k) {
return k==0 || k==n ? 1 : binomial(n-1,k-1) + binomial(n-1,k);
}
```

```
void main() {
cout << endl;
for (int i = 0; i < NROWS; i++) {
   int j;
   for (j = 0; j<(NROWS-i)*3+1; j++) cout << " ";
   for (j = 0; j <= i; j++)
      cout << setw(6) << binomial(i,j);
   cout << endl;
   }
}
```

---

### a480bina.cpp

```
/*************************************************************************/
/*                                                                       */
/*   Visualizza la rappresentazione in base 2 di un intero "n".          */
/*                                                                       */
/*************************************************************************/
#include <iostream.h>


void binary (int n) {
if (n > 0) {
   binary(n / 2);
   cout << n % 2;
   }
}

void main() {
cout << "Conversione di numeri interi positivi da decimale a binario\n";
cout << "-------------------------------------------------------------\n";
int n;
do {
   cout << "Introdurre numero (<=0 per finire): ";
   cin >> n;
   if (n > 0)
      binary(n);
   cout << endl;
   } while (n > 0);
}
```

---

### a485enum.cpp

```
// nuovi elementi: enum
// nuovi concetti: tipo enumerato
#include <iostream.h>

enum Color {red, orange, yellow, green, blue, violet};

main()
{
Color x = blue;
cout << "x = " << x << endl;
return 0;
}

/* Output:
x = 4
*/
```

## a490enum.cpp

```
// nuovi concetti: interi ed enumerati
#include <iostream.h>

enum Color {red, orange, yellow, green, blue, violet};

main()
{
Color x;
int y;
y = yellow;
cout << "y = " << y << endl;
x = Color(3);
cout << "x = " << x << endl;
y = green + 2;
cout << "y = " << y << endl;
return 0;
}

/* Output:
y = 2
x = 3
y = 5
*/
```

## a495aren.cpp

```
// nuovi concetti: tipo enum come indice di un array
#include <iostream.h>

void main() {
  enum Day       {lun,  mar,  mer,  gio,  ven,  sab,  dom};
  float high[] = {88.3, 95.0, 91.2, 89.9, 91.4, 92.5, 86.7};
  for (Day day = lun; day <= dom; int(day)++)
    cout << "La temperatura massima del giorno " << day
         << " e' stata " << high[day] << endl;
}
```

## a500bool.cpp

```
// nuovi concetti: definizione di un tipo Boolean
#include <iostream.h>

enum Boolean {falso, vero};

void getInfo(Boolean& coniugato, Boolean& ConiugeLavora);

void main() {
Boolean coniugato, ConiugeLavora;
getInfo(coniugato, ConiugeLavora);
if (coniugato) {
   cout << "Sei coniugato/a.\n";
   cout << "Il coniuge "
        << (ConiugeLavora ? "non" : "")
        << " e' a carico.\n";
   }
else cout << "Sei celibe/nubile.\n";
}
```

```
void getInfo(Boolean& coniugato, Boolean& ConiugeLavora) {
char ans;
cout << "Sei coniugato/a?(S/N) ";  cin >> ans;
coniugato = Boolean(ans == 's'  ||  ans == 'S');
if (coniugato) {
  cout << "Il coniuge e' a carico?(S/N) ";  cin >> ans;
  ConiugeLavora = Boolean(ans != 's'  &&  ans != 'S');
  }
else ConiugeLavora = falso;
}
```

---

### a505enum.cpp

```
#include <iostream.h>

void main() {
enum {a,b,c,d,e,f};
cout << b+d << endl;
}
```

---

### a510enum.cpp

```
#include <iostream.h>

enum Vocali {a=1,e,i,o,u};
void main() {
Vocali vocale = u;
cout << vocale << endl;
}
```

---

### a515tdef.cpp

```
// nuovi elementi: typedef
#include <iostream.h>

void main() {
const int DIM = 6;
typedef int Vettore[DIM];
int i;
Vettore a;

cout << "Introduci " << DIM << " numeri interi.\n";
for (i=0; i<DIM; i++)
   cin >> a[i];
cout << endl;
for (i=DIM; i>0; )
   cout << a[--i] << endl;
}
```

---

### a520matr.cpp

```
#include <iostream.h>

void main() {
const int DIM = 3;
typedef int Matrice[DIM][DIM];
```

```
int i, j;
Matrice m, t;

cout << "Introdurre matrice (" << DIM << 'x' << DIM << ")\n";
for (i=0; i<DIM; i++) {
   for (j=0; j<DIM; j++)
      cin >> m[i][j];
   }
for (i=0; i<DIM; i++)
   for (j=0; j<DIM; j++)
      t[i][j] = m[j][i];
cout << "\nMatrice introdotta:\n";
for (i=0; i<DIM; i++) {
   for (j=0; j<DIM; j++)
      cout << m[i][j] << " ";
   cout << endl;
   }
cout << "\nMatrice trasposta:\n";
for (i=0; i<DIM; i++) {
   for (j=0; j<DIM; j++)
      cout << t[i][j] << " ";
   cout << endl;
   }
}
```

---

### a525cin.cpp

```
// nuovi concetti: (cin)
#include <iostream.h>

void main() {
int n, sum=0;
cin >> n;
while (cin) {
   sum += n;
   cin >> n;
        }
cout << "somma = " << sum << endl;
}
```

---

### a530cing.cpp

```
// nuovi elementi: cin.get()
#include <iostream.h>

void main() {
char c;
while (cin.get(c))
        cout << c;
cout << endl;
}
```

---

### a535cget.cpp

```
#include <iostream.h>

void main() {
char ch;
```

```
int count = 0;
while (cin.get(ch))
        if (ch == 'e') ++count;
cout << "La lettera \"e\" e' stata contata " << count << " volte.\n";
}
```

---

### a540cput.cpp

```
// nuovi elementi: cout.put()
#include <iostream.h>
#include <ctype.h>

void main() {
char ch, pre = '\n';
while (cin.get(ch)) {
   if (pre == ' ' || pre == '\t' || pre == '\n')
      cout.put(toupper(ch));
   else
      cout.put(ch);
   pre = ch;
   }
}
```

---

### a545cing.cpp

```
// nuovi elementi: cin.get() senza parametri
#include <iostream.h>

void main() {
char c;
while ((c = cin.get()) != EOF)
        cout << c;
cout << endl;
}
```

---

### a550eof.cpp

```
#include <iostream.h>

void main() {
cout << "EOF = " << EOF << endl;
}

/* Output:
EOF = -1
*/
```

---

### a555cing.cpp

```
// nuovi elementi: cin.get() con 2 parametri
#include <iostream.h>

void main() {
char buffer[80];
cin.get(buffer,8);
```

```
cout << "[" << buffer << "]\n";
cin.get(buffer,sizeof(buffer));
cout << "[" << buffer << "]\n";
}
```

---

### a560peek.cpp

```
// nuovi elementi: cin.peek(), cin.ignore()
#include <iostream.h>

int nextInt();

void main() {
int m = nextInt(), n = nextInt();
cin.ignore(80,'\n');
cout << m << " + " << n << " = " << m+n << endl;
}

int nextInt() {
char ch;
int n = 0;
while ((ch = cin.peek()) != EOF)
   if (ch >= '0' && ch <= '9') {
      cin >> n;
      break;
      }
   else cin.get(ch);
return n;
}
```

---

### a565rice.cpp

```
// nuovi concetti: algoritmo di ricerca sequenziale
#include <iostream.h>
#include <stdlib.h>

void main() {
const int n = 10;
int i, x, a[n];

for (i=0; i<n; i++) {
   a[i] = random(20);     // genera un numero intero a caso tra 0 e 19
   cout << a[i] << " ";
   }

cout << "\nIntroduci un numero intero: ";
cin >> x;

for (i = 0; i < n && a[i] != x; i++)
    // il corpo del for e' vuoto
    // se  i >= n,  l'espressione  a[i] != x  non viene valutata
    ;
if (i < n)
   cout << "Trovato al posto " << i+1 << endl;
else
   cout << "Non trovato" << endl;
}
```

---

### a570rice.cpp

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

void main() {
const int n = 10;
int i, x, a[n+1];

for (i=0; i<n; i++)  {
   cout << (a[i] = random(20)) << " ";
   }

cout << "\nIntroduci un numero intero: ";
cin >> x;
a[n] = x;
for (i=0; a[i]!=x; i++);   // il ciclo termina quando  a[i] == x
if (i<n)
   cout << "Trovato al posto " << (i+1) << endl;
else
   cout << "Non trovato" << endl;
}
```

---

**a575rice.cpp**

```
// nuovi concetti: algoritmo di ricerca binaria
#include <iostream.h>

void main() {
const int n = 200;
int i, max, min, med, x, trovato=0, a[n];

for (i=0; i<n; i++)  {
   a[i] = i+1;
   cout << a[i] << (i%10 == 9 ? "\n" : " ");
   }

cout << "\nIntroduci un numero intero: ";
cin >> x;
max = n-1;
min = 0;

while (!trovato && max>=min) {
   med = (max+min)/2;
   if (a[med] == x)
      trovato = 1;
   else if (a[med] > x)
        max = med-1;
   else min = med+1;
   }

if (trovato)
   cout << x << " si trova nell'insieme alla posizione " << (med+1) << endl;
else
   cout << x << " non si trova nell'insieme" << endl;
}
```

---

**a580rice.cpp**

```
// nuovi concetti: ricerca binaria ricorsiva
#include <iostream.h>


int ricerca (int [], int, int, int, int&);
```

```
void main() {
const N = 1000;
int i, x, pos, a[N];

for (i=0; i<N; i++)  {
   a[i] = i+1;
   cout << a[i] << " ";
   }
cout << endl;

cout << "Introdurre un numero intero: ";
cin >> x;
if (ricerca(a,0,N-1,x,pos))
   cout << x << " si trova nell'insieme alla posizione " << (pos+1) << endl;
else
   cout << x << " non presente nell'insieme!" << endl;
}

int ricerca (int a[], int min, int max, int val, int& pos) {
if (min > max)
   return 0;
pos = (max+min) / 2;
if (a[pos] == val)
   return 1;
return (a[pos] > val) ? ricerca(a,min,pos-1,val,pos) :
                        ricerca(a,pos+1,max,val,pos);
}
```

---

**a585ordi.cpp**

```
// nuovi elementi: cin.get()
// nuovi concetti: algoritmo di ordinamento ingenuo
#include <iostream.h>
#include <stdlib.h>

void main() {
const int n=400;
int i, j, a[n];

randomize();
// Genera array
for (i=0; i<n; i++)  a[i]=i+1;
// Mescola gli elementi dell'array
for (i=0; i<n; i++) {
   j = random(n);
   int t = a[i];
   a[i] = a[j];
   a[j] = t;
   }
// Visualizza array mescolato
for (i=0; i<n; i++)
   cout << a[i] << (i%20 == 19 ? "\n" : " ");
// Ordina array
cout << "\nInizio ordinamento\n";
for (i=0; i<n-1; i++)
   for (j=i+1; j<n; j++)
      if (a[j] < a[i]) {
         int t = a[i];
         a[i] = a[j];
         a[j] = t;
         }
cout << "Fine ordinamento.  Premere Invio...\n";
cin.get();
// Visualizza array ordinato
for (i=0; i<n; i++)
```

```
   cout << a[i] << (i%20 == 19 ? "\n" : " ");
}
```

---

## a590bubl.cpp

```
// nuovi concetti: algoritmo di ordinamento "Bubble Sort"
#include <iostream.h>
#include <stdlib.h>

void main() {
const int n=400;
int i, j, sup, ultimo_scambiato, t, a[n];

randomize();
for (i=0; i<n; i++)  a[i]=i+1;
for (i=0; i<n; i++) {
   t = a[i];
   j = random(n);
   a[i] = a[j];
   a[j] = t;
   }
for (i=0; i<n; i++)
   cout << a[i] << (i%20 == 19 ? "\n" : " ");
cout << "\nInizio ordinamento\n";
sup = n-1;
while (sup >= 0) {
   ultimo_scambiato = -1;
   for (i=0; i<sup; i++)
      if (a[i]>a[i+1]) {
         t = a[i];
         a[i] = a[i+1];
         a[i+1] = t;
         ultimo_scambiato = i;
         }
   sup = ultimo_scambiato;
   }
cout << "Fine ordinamento.  Premere Invio...\n";
cin.get();
for (i=0; i<n; i++)
   cout << a[i] << (i%20 == 19 ? "\n" : " ");
}
```

---

## a595quik.cpp

```
// nuovi concetti: algoritmo "quick sort"
#include <iostream.h>

int X[] = {54,6,26,73,1,69,36,70,14,76,81,85,99,8,16,38,45};
int n = sizeof(X)/sizeof(int);

void mostra_sequenza(char intestazione[])
{
cout << "\n" << intestazione;
for (int i = 0; i < n; i++)
   cout << " " << X[i];
cout << "\n";
}

void quick_sort(int primo, int ultimo)
{
int temp;          // variabile temporanea per lo scambio
if (primo < ultimo)
```

```
   {
//prendiamo come pivot il primo elemento della sequenza
   int pivot = X[primo];
   int i = primo;
   int j = ultimo;
   while (i < j)
      {
// cerca, partendo da sinistra, il primo elemento > pivot
      while (i < ultimo && X[i] <= pivot)
         i++;
// cerca, partendo da destra, il primo elemento < pivot
      while (j > primo && X[j] >= pivot)
         j--;
      if (i < j) //scambio(X[i],X[j])
         {
         temp = X[i];
         X[i] = X[j];
         X[j] = temp;
         }

      }
   //scambio(X[j],X[primo])
   temp = X[primo];
   X[primo] = X[j];
   X[j] = temp;

//applico ricorsivamente quick sort alle due parti
   quick_sort(primo, j-1);
   quick_sort(j+1, ultimo);
   }
}

void main(void)
{
mostra_sequenza("Sequenza da ordinare:\n");
quick_sort(0,n-1);
mostra_sequenza("Sequenza ordinata con 'quick sort':\n");
}
```

---

### a600size.cpp

```
// nuovi elementi: sizeof(), double, long double
#include <iostream.h>

main()
{
cout << "Numero di byte occupati\n";
cout << "\t          char: " <<           sizeof(char) << endl;
cout << "\t         short: " <<          sizeof(short) << endl;
cout << "\t           int: " <<            sizeof(int) << endl;
cout << "\t          long: " <<           sizeof(long) << endl;
cout << "\t unsigned char: " <<  sizeof(unsigned char) << endl;
cout << "\tunsigned short: " << sizeof(unsigned short) << endl;
cout << "\t  unsigned int: " <<   sizeof(unsigned int) << endl;
cout << "\t unsigned long: " <<  sizeof(unsigned long) << endl;
cout << "\t   signed char: " <<    sizeof(signed char) << endl;
cout << "\t         float: " <<          sizeof(float) << endl;
cout << "\t        double: " <<         sizeof(double) << endl;
cout << "\t   long double: " <<    sizeof(long double) << endl;
return 0;
}
```

---

### a605flot.cpp

```
// nuovi elementi: float.h
// nuovi concetti: rappresentazione del tipo float
#include <iostream.h>
#include <float.h>

main()
{
int fbits = 8 * sizeof(float);
cout << "float occupa:\t" << fbits << " bit:\n\t\t"
     <<         FLT_MANT_DIG - 1 << " bit per la mantissa\n\t\t "
     <<     fbits - FLT_MANT_DIG << " bit per l'esponente\n\t\t "
     <<                        1 << " bit per il segno\n"
     << "        per ottenere: " << FLT_DIG << " cifre significative\n"
     << "   col valore minimo: " << FLT_MIN << endl
     << "e col valore massimo: " << FLT_MAX << endl;
return 0;
}
```

---

### a610punt.cpp

```
// nuovi concetti: indirizzo di una variabile
#include <iostream.h>

void main() {
int n = 33;
cout << "n = " << n << endl;
cout << "&n = " << &n << endl;
}

/* Output:
n = 33
&n = 0x48172890
*/
```

---

### a615alia.cpp

```
// nuovi concetti: riferimento indipendente
#include <iostream.h>

void main() {
int n = 33;
int& r = n;    // r e' adesso un alias di n
cout << "n = " << n << ", r = " << r << endl;
--n;
cout << "n = " << n << ", r = " << r << endl;
r *= 2;
cout << "n = " << n << ", r = " << r << endl;
}

/* Output:
n = 33, r = 33
n = 32, r = 32
n = 64, r = 64
*/
```

---

### a620alia.cpp

```
#include <iostream.h>
```

```
void main() {
int n = 33;
int& r = n;
cout << "&n = " << &n << ", &r = " << &r << endl;
}

/* Output:
&n = 0x0e272892, &r = 0x0e272892
*/
```

---

### a625indi.cpp

```
// nuovi concetti: variabile puntatore
#include <iostream.h>

void main() {
int n = 33;
int* p = &n;
cout << "n = " << n << ", &n = " << &n << ", p = " << p << endl;
cout << "&p = " << &p << endl;
}

/* Output:
n = 33, &n = 0x0edf289e, p = 0x0edf289e
&p = 0x0edf289a
*/
```

---

### a630inin.cpp

```
#include <iostream.h>

void main() {
int n = 33;
int* p = &n;
cout << "*p = " << *p << endl;
}

/* Output:
*p = 33
*/
```

---

### a635rife.cpp

```
#include <iostream.h>

void main() {
int n = 33;
int* p = &n;      // p punta a n
int& r = *p;      // r e' un riferimento per n
cout << "r = " << r << endl;
}

/* Output:
r = 33
*/
```

## a640furi.cpp

```
// nuovi concetti: funzione che restituisce un riferimento
#include <iostream.h>

int& max(int& m, int& n) {  // max restituisce un riferimento a int
return (m > n ? m : n);     // m e n sono riferimenti non locali
}

void main() {
int m = 44, n = 22;
cout << m << ", " << n << ", " << max(m,n) << endl;
max(m,n) = 55;              // cambia il valore di m da 44 a 55
cout << m << ", " << n << ", " << max(m,n) << endl;
}

/* Output:
44, 22, 44
55, 22, 55
*/
```

## a645puar.cpp

```
// nuovi concetti: inizializzazione di un array
#include <iostream.h>

void main() {
double a[4] = {22.2, 44.4, 66.6, 88.8};
cout << a << endl;
cout << *a << endl;
cout << *(a+1) << endl;
for (int i = 0; i < 4; i++)
   cout << "a[" << i << "] = " << a[i] << endl;
}

/* Output:
ab0x8f5effd6
22.2
44.4
a[0] = 22.2
a[1] = 44.4
a[2] = 66.6
a[3] = 88.8
*/
```

## a650fuin.cpp

```
#include <iostream.h>

float& component(float* v, int k) {
return v[k-1];
}

void main() {
float v[4];
for (int k = 1; k <= 4; k++)
        component(v,k) = 1.0/k;    // equivale a  v[k-1] = 1.0/k;
for (int i = 0; i < 4; i++)
        cout << "v[" << i << "] = " << v[i] << endl;
}
```

```
/* Output:
v[0] = 1
v[1] = 0.5
v[2] = 0.333333
v[3] = 0.25
*/
```

---

### a655aray.cpp

```
// nuovi concetti: aritmetica dei puntatori
#include <iostream.h>

void main() {
short a[] = {22, 33, 44};
int size = sizeof(a) / sizeof(short);
cout << "a = " << a << endl;
cout << "size = " << size << endl;
cout << "sizeof(short) = " << sizeof(short) << endl;
short* end = a + size;
short sum = 0;
for (short* p = a; p < end; p++) {
   sum += *p;
   cout << "\t p = " << p;
   cout << "\t *p = " << *p;
   cout << "\t sum = " << sum << endl;
   }
cout << "end = " << end << endl;
}
```

---

### a660indx.cpp

```
#include <iostream.h>

void main() {
short a[] = {22, 33, 44, 55, 66};
cout << "a = " << a << ", *a = " << *a << endl;
for (short* p = a; p < a+5; p++)
        cout << "p = " << p << ", *p = " << *p << endl;
}
```

---

### a665sequ.cpp

```
#include <iostream.h>

short* loc(short* a1, short* a2, int n1, int n2) {
short* end1 = a1 + n1 - n2;
for (short* p1 = a1; p1 <= end1; p1++)
   if (*p1 == *a2) {
      int j;
      for (j = 0; j < n2; j++)
         if (p1[j] != a2[j]) break;
      if (j == n2) return p1;
      }
return NULL;
}

void main() {
```

```
short a1[9] = {11, 11, 11, 11, 11, 22, 33, 44, 55};
short a2[5] = {11, 11, 11, 22, 33};
cout << "L'array a1 inizia all'indirizzo   \t" << a1 << endl;
cout << "L'array a2 inizia all'indirizzo   \t" << a2 << endl;
short* p = loc(a1, a2, 9, 5);
if (p) {
   cout << "L'array a2 trovato all'indirizzo   \t" << p << endl;
   for (int i = 0; i < 5; i++)
      cout << '\t' << &p[i] << ": " << p[i]
           << '\t' << &a2[i] << ": " << a2[i] << endl;
   }
else cout << "Non trovato.\n";
}
```

---

**a670new.cpp**

```
// nuovi elementi: new, delete
// nuovi concetti: allocazione dinamica della memoria
#include <iostream.h>

int main()
{
    int * pi = new int;         // alloca spazio per un int
    *pi = 1001;                 // memorizza un valore in quello spazio
    cout << "int: ";
    cout << "*pi = " << *pi << ", pi = " << pi << "\n";
    cout << "sizeof pi = " << sizeof pi;
    cout << ", sizeof *pi = " << sizeof *pi << "\n";
    delete pi;                  // libera lo spazio allocato

    double * pd = new double;   // alloca spazio per un double
    *pd = 10000001.0;           // memorizza un valore in quello spazio
    cout << "double: ";
    cout << "*pd = " << *pd << ", pd = " << pd << "\n";
    cout << "sizeof pd = " << sizeof pd;
    cout << ", sizeof *pd = " << sizeof *pd << "\n";
    delete pd;                  // libera lo spazio allocato

    return 0;
}
```

---

**a675ardi.cpp**

```
// nuovi elementi: delete
// nuovi concetti: riferimento a puntatore
#include <iostream.h>

void get(double*&, int&);
void print(double*, int);

void main() {
double* a;
int n;
get(a,n);
print(a,n);
delete [] a;
get(a,n);
print(a,n);
delete [] a;
}

void get(double*& a, int& n) {  // a e' un puntatore passato per riferimento
```

```
cout << "Introduci numero di elementi: ";
cin >> n;
a = new double[n];
cout << "Introduci elementi, uno per linea:\n";
for (int i = 0; i < n; i++) {
   cout << i+1 << ": ";
   cin >> a[i];
        }
}

void print(double* a, int n) {
for (int i = 0; i < n; i++)
        cout << a[i] << " ";
cout << endl;
}
```

---

**a680ordi.cpp**

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

void sort(int [], int);

void main() {
const int n=400;
int i, a[n];

randomize();
for (i=0; i<n; i++)  a[i]=i+1;
for (i=0; i<n; i++) {
   int j = random(n);
   int t = a[i];
   a[i] = a[j];
   a[j] = t;
   }
for (i=0; i<n; i++)
   cout << a[i] << (i%20 == 19 ? "\n" : " ");
cout << "\nInizio ordinamento\n";
sort(a,n);
cout << "Fine ordinamento.  Premere Invio...\n";
cin.get();
for (i=0; i<n; i++)
   cout << a[i] << (i%20 == 19 ? "\n" : " ");
}

void sort(int a[], int n) {
for (int i=0; i<n-1; i++)
   for (int j=i+1; j<n; j++)
      if (a[j] < a[i]) {
         int t = a[i];
         a[i] = a[j];
         a[j] = t;
         }

}
```

---

**a685ordi.cpp**

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

void sort(int* [], int);
```

```
void main() {
const int n=400;
int i, *a[n];

randomize();
for (i=0; i<n; i++)  a[i]=new int(i+1);
for (i=0; i<n; i++)
   cout << *a[i] << (i%20 == 19 ? "\n" : " ");
cout << "Premere Invio...\n";
cin.get();
for (i=0; i<n; i++) {
   int j = random(n);
   int* t = a[i];
   a[i] = a[j];
   a[j] = t;
   }
for (i=0; i<n; i++)
   cout << *a[i] << (i%20 == 19 ? "\n" : " ");
cout << "\nInizio ordinamento\n";
sort(a,n);
cout << "Fine ordinamento.  Premere Invio...\n";
cin.get();
for (i=0; i<n; i++)
   cout << *a[i] << (i%20 == 19 ? "\n" : " ");
}

void sort(int* a[], int n) {
for (int i=0; i<n-1; i++)
   for (int j=i+1; j<n; j++)
      if (*a[j] < *a[i]) {
         int* t = a[i];
         a[i] = a[j];
         a[j] = t;
         }

}
```

---

### a690pafu.cpp

```
// nuovi concetti: funzione come parametro
#include <iostream.h>

int sum(int f(int), int);
int square(int);
int cube(int);

void main() {
cout << sum(square,4) << endl;
cout << sum(cube,4) << endl;
}

int sum(int f(int), int n) {
int s = 0;
for (int i = 1; i <= n; i++)
   s += f(i);
return s;
}

int square(int k) {
return k*k;
}

int cube(int k) {
return k*k*k;
}
```

```
/* Output:
30
100
*/
```

---

### a695strn.cpp

```
// nuovi concetti: stringa come puntatore costante a char
#include <iostream.h>
#include <string.h>

void main() {
char animale[] = "cavallo";
char * pesce = "tonno";

cout << animale << ", " << strlen(animale) << endl;
cout << pesce << ", " << strlen(pesce) << endl;
cout << animale[2] << endl;
cout << pesce[1] << endl;
}
```

---

### a700inas.cpp

```
// nuovi concetti: inizializzazione di array di stringhe
#include <iostream.h>

void main() {
char* name[] = { "George Washington",
                 "John Adams",
                 "Thomas Jefferson" };
cout << "I nomi sono:\n";
for (int i = 0; i < 3; i++)
   cout << '\t' << i << ". [" << name[i] << "]" << endl;
}
```

---

### a705strc.cpp

```
// nuovi elementi: strchr(), strrchr(), strstr()
// nuovi concetti: localizzazione di caratteri e sottostringhe
//                 all'interno di una stringa
#include <iostream.h>
#include <string.h>

void main() {
char s[] = "The Mississippi is a long river.";
cout << "s = \"" << s << "\"\n";
char* p = strchr(s,' ');
cout << "strchr(s,' ') punta a s[" << p-s << "].\n";
p = strchr(s,'s');
cout << "strchr(s,'s') punta a s[" << p-s << "].\n";
p = strrchr(s,'s');
cout << "strrchr(s,'s') punta a s[" << p-s << "].\n";
p = strstr(s,"is");
cout << "strstr(s,\"is\") punta a s[" << p-s << "].\n";
p = strstr(s,"isi");
if (p == NULL) cout << "strstr(s,\"isi\") restituisce NULL\n";
}
```

## a710stok.cpp

```
// nuovi concetti: isolare le parole da una frase
// nuovi elementi: strtok()
#include <iostream.h>
#include <string.h>

void main() {
char s[] = "Today's date is March 12, 1995.";
char* p;
cout << "La stringa e': [" << s << "]\nI token sono:\n";
p = strtok(s," ");
while (p) {
   cout << "\t[" << p << "]\n";
   p = strtok(NULL, " ");
   }
cout << "Ora la stringa e': [" << s << "]\n";
}
```

## a715stpb.cpp

```
// nuovi elementi: strpbrk()
#include <iostream.h>
#include <string.h>

void main() {
char s[] = "The Mississippi is a long river.";
cout << "s = \"" << s << "\"\n";
char* p = strpbrk(s, "nopqr");
cout << "strpbrk(s, \"nopqr\") punta a s[" << p - s << "].\n";
p = strpbrk(s, "NOPQR");
if (p == NULL) cout << "strpbrk(s, \"NOPQR\") restituisce NULL.\n";
}
```

## a720time.cpp

```
// nuovi elementi: time.h
#include <iostream.h>
#include <time.h>

void main() {
clock_t c;
time_t t;
char* s;
double x = 3.14159265358979;
cout << "CLOCKS_PER_SEC = " << CLOCKS_PER_SEC << endl;
for (long int i = 0; i < 1000000; i++)
        x = 1 + x/(1+x);
c = clock();
cout << "clock() = " << c << endl;
cout << "secondi = " << c/CLOCKS_PER_SEC << endl;
t = time(NULL);
cout << "time(NULL) = " << t << endl;
s = ctime(&t);
cout << "data corrente = " << s << endl;
}
```

## a725cplx.cpp

```
// nuovi elementi: struct
#include <iostream.h>
#include <math.h>

struct Complesso {
       float re, im;
       };


void leggi (Complesso& c) {
cout << "    parte reale: ";
cin >> c.re;
cout << "    parte immag: ";
cin >> c.im;
}

void scrivi (Complesso c) {
cout << c.re << " ";
if (c.im)
   if (c.im > 0)
      cout << " +i" << c.im;
   else
      cout << " -i" << -c.im;
cout << endl;
}

Complesso add (Complesso c1, Complesso c2) {
Complesso sum;
sum.re = c1.re + c2.re;
sum.im = c1.im + c2.im;
return sum;
}

Complesso sub (Complesso c1, Complesso c2) {
Complesso diff;
diff.re = c1.re - c2.re;
diff.im = c1.im - c2.im;
return diff;
}

Complesso mul (Complesso c1, Complesso c2) {
Complesso prod;
prod.re = c1.re * c2.re - c1.im * c2.im;
prod.im = c1.re * c2.im + c1.im * c2.re;
return prod;
}

float modcom (Complesso c) {
return sqrt ( c.re * c.re + c.im * c.im );
}

Complesso divi (Complesso c1, Complesso c2) {
Complesso divis;
float m = modcom(c2);
divis.re = ( c1.re * c2.re + c1.im * c2.im) / m;
divis.im = ( c1.im * c2.re - c1.re * c2.im) / m;
return divis;
}

void main() {

Complesso a, b;

cout << "Operazioni elementari con numeri complessi\n\n";
cout << "primo numero\n";
```

```
leggi(a);
cout << "secondo numero\n";
leggi(b);
cout << "\nprimo numero    = ";
scrivi(a);
cout << "secondo numero  = ";
scrivi(b);
cout << "somma           = ";
scrivi(add(a,b));
cout << "sottrazione     = ";
scrivi(sub(a,b));
cout << "moltiplicazione = ";
scrivi(mul(a,b));
cout << "divisione       = ";
if (modcom(b))
   scrivi(divi(a,b));
else
   cout << " divisione per zero\n";
}
```

---

**a730cplx.cpp**

```
// nuovi concetti: incapsulamento
#include <iostream.h>
#include <math.h>

struct Complesso {
      float re, im;
      void leggi();
      void scrivi();
      float modcom() {return sqrt(re * re + im * im);}
      };


void Complesso::leggi() {
cout << "    parte reale: ";
cin >> re;
cout << "    parte immag: ";
cin >> im;
}

void Complesso::scrivi() {
cout << re << " ";
if (im)
        if (im > 0)
                cout << " +i" << im;
        else
                cout << " -i" << -im;
cout << endl;
}


Complesso add (Complesso c1, Complesso c2) {
Complesso sum;
sum.re = c1.re + c2.re;
sum.im = c1.im + c2.im;
return sum;
}

Complesso sub (Complesso c1, Complesso c2) {
Complesso diff;
diff.re = c1.re - c2.re;
diff.im = c1.im - c2.im;
return diff;
}
```

```
Complesso mul (Complesso c1, Complesso c2) {
Complesso prod;
prod.re = c1.re * c2.re - c1.im * c2.im;
prod.im = c1.re * c2.im + c1.im * c2.re;
return prod;
}

Complesso divi (Complesso c1, Complesso c2) {
Complesso divis;
float m = c2.modcom();
divis.re = ( c1.re * c2.re + c1.im * c2.im) / m;
divis.im = ( c1.im * c2.re - c1.re * c2.im) / m;
return divis;
}

void main() {

Complesso a, b;

cout << "Operazioni elementari con numeri complessi\n\n";
cout << "primo numero\n";
a.leggi();
cout << "secondo numero\n";
b.leggi();
cout << endl;
cout << "primo numero    = ";
a.scrivi();
cout << "secondo numero  = ";
b.scrivi();
cout << "somma           = ";
add(a,b).scrivi();
cout << "sottrazione     = ";
sub(a,b).scrivi();
cout << "moltiplicazione = ";
mul(a,b).scrivi();
cout << "divisione       = ";
if (b.modcom())
   divi(a,b).scrivi();
else
   cout << " divisione per zero" << endl;
}
```

---

### a735ofst.cpp

```
// nuovi elementi: <fstream.h>, ofstream, close
// nuovi concetti: scrittura su file
#include <fstream.h>

void main() {
ofstream fout("miofile.txt");    // apro il file
fout << "Questa e' l'unica riga del file." << endl;  // scrivo sul file
fout.close();                    // chiudo il file
}
```

---

### a740ifst.cpp

```
// nuovi elementi: ifstream
// nuovi concetti: lettura da file
#include <fstream.h>

void main() {
ifstream fin("miofile.txt");    // apro il file
```

```
char testo[80];
fin.getline(testo, 80);         // leggo una riga dal file
fin.close();                    // chiudo il file
cout << testo << endl;          // visualizzo la riga letta
}
```

---

### a745nint.cpp

```
// nuovi concetti: fin.peek()
#include <fstream.h>

int nextInt();

ifstream fin("miofile.txt");    // apro il file

void main() {
int m = nextInt(), n = nextInt();
fin.close();
cout << m << " + " << n << " = " << m+n << endl;
}

int nextInt() {
char ch;
int n = 0;
while ((ch = fin.peek()) != EOF)
   if (ch >= '0' && ch <= '9') {
      fin >> n;
      break;
      }
   else fin.get(ch);
return n;
}
```

---

### a750ofst.cpp

```
// nuovi elementi: cerr
#include <fstream.h>
#include <stdlib.h>    // -> exit()

void main() {
ofstream outfile("voti.dat");
if (!outfile) {
        cerr << "Errore: file output non puo' venire aperto.\n";
        exit(1);
        }
char matricola[9], name[16];
int voto;
cout << "nome matricola voto (^Z per finire)\n";
cout << "\t1: ";
int n = 1;
while (cin >> name >> matricola >> voto) {
   outfile << name << " " << matricola << " " << voto << endl;
   cout << '\t' << ++n << ": ";
        }
outfile.close();
}
```

---

### a755ifst.cpp

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

void main() {
ifstream infile("voti.dat");
if (!infile) {
        cerr << "Errore: file input non puo' venire aperto.\n";
        exit(1);
        }
char nmatr[9], name[16];
int voto, sum = 0, count = 0;
while (infile >> name >> nmatr >> voto) {
   cout << name << "," << nmatr << "," << voto << endl;
   sum += voto;
   ++count;
        }
infile.close();
cout << "La votazione media e' " << float(sum)/count << endl;
}
```

---

### a760argv.cpp

```
// nuovi elementi: argc, argv
#include <iostream.h>

void main(int argc, char* argv[]) {
cout << "Il comando " << argv[0];
if (argc < 2)    // ovvero,  argc == 1
   cout << " non ha argomenti!" << endl;
else {
   if (argc == 2)
      cout << " ha il seguente argomento:\n";
   else  // argc > 2
      cout << " ha i seguenti " << argc-1 << " argomenti:\n";
   for (int i=1; i<argc; i++)
      cout << argv[i] << endl;
   }
}
```

---

### a765piuf.cpp

```
// nuovi elementi: open()
#include <fstream.h>
#include <stdlib.h>

int main(int argc, char * argv[])
{
if (argc == 1)             // termina se non ci sono argumenti
   {
   cerr << "Uso: " << argv[0] << " <nomifile>\n";
   exit(1);
   }

ifstream fin;              // apri stream
long conto;
long totale = 0;
char ch;

for (int file = 1; file < argc; file++)
   {
   fin.open(argv[file]);   // connetti stream a argv[file]
   if (!fin)
```

```
      {
      cerr << "Non sono riuscito ad aprire il file " << argv[file] << "\n";
      continue;
      }
   for (conto = 0; fin.get(ch); conto++);
   cout << conto << " caratteri in " << argv[file] << "\n";
   totale += conto;
   fin.close();            // disconnetti file
   }
cout << totale << " caratteri in totale nei file\n";

return 0;
}
```

---

### a770fapp.cpp

```
// nuovi elementi: ios::app
// nuovi concetti: aggiunta a un file
#include <fstream.h>

void main() {
ofstream fout("miofile.txt", ios::app);
         // il file e' aperto in modalita' append
fout << "Questa e' l'unica riga del file?" << endl;
fout.close();
}
```

---

### a775rubr.cpp

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

const char * file = "rubrica.txt";
const int Lun = 40;

void acapo() {
while (cin.get() != '\n')
   continue;
}

void main() {
char ch;
ifstream fin(file);
if (fin) {
   cout << "Questo e' il contenuto del file " << file << ":\n";
   while (fin.get(ch))
      cout << ch;
   }
fin.close();
ofstream fout(file, ios::app);
if (!fout) {
   cerr << "Non e' possibile aprire il file " << file << " in output.\n";
   exit(1);
   }
char nome[Lun];
do {
   cout << "Introdurre un nome (riga vuota per terminare):\n";
   cin.get(nome,Lun);
   if (nome[0]) {
      acapo();
      fout << nome << "\t";
      cout << "Numero telefonico: ";
```

```
      cin.get(nome,Lun);
      acapo();
      fout << nome << "\n";
      }
   } while (nome[0]);
fout.close();

fin.open(file);
if (fin) {
   cout << "Questo e' il nuovo contenuto del file " << file << ":\n";
   while (fin.get(ch))
      cout << ch;
   }
fin.close();
}
```

---

### a780fstr.cpp

```
// nuovi elementi: fstream, ios::in, ios::out, |, seekp(), ios::cur
// nuovi concetti: lettura-scrittura su file
// ATTENZIONE: ACCERTARSI CHE I FILE FORNITI COME ARGOMENTO POSSANO ESSERE MODIFICATI
#include <fstream.h>
#include <stdlib.h>
#include <ctype.h>
#include <conio.h>

void main(int argc, char* argv[]) {
if (argc < 2) {
   cout << "Bisogna fornire il nome di un file (modificabile) come argomento!" << endl;
   exit(1);
   }
fstream iofile(argv[1], ios::in | ios::out);
if (!iofile) {
   cerr << "Errore: file in/out non puo' venire aperto.\n";
   exit(2);
   }
else {
   cout << "Il file " << argv[1] << " esiste gia': posso modificarlo? (s/N)";
   char ch = toupper(getche());
   if (ch != 'S') exit(3);
   }
cout << "\nSto modificando il file " << argv[1] << ".....\n";
char c;
while ((c = iofile.get()) != EOF)
   if (islower(c)) {
      iofile.seekp(-1, ios::cur);  // torna indietro di un byte
      c = toupper(c);
      iofile.put(c);
      }
iofile.close();
cout << "Il file " << argv[1] << " e' stato modificato!" << endl;
}
```

---

### a785read.cpp

```
// nuovi elementi: cin.read()
#include <iostream.h>

void main() {
char buffer[] = "??????????????????????";
cin.read(buffer,8);
cout << "[" << buffer << "]" << endl;
```

```
cin.read(buffer,4);
cout << "[" << buffer << "]" << endl;
}
```

---

### a790writ.cpp

```
// nuovi elementi: cout.write()
#include <iostream.h>

void main() {
cout.write("ABCDEFGHIJKLMNOPQRSTUVWXYZ", 8);
cout << endl;
char * stringa = "0123456789";
cout.write(stringa,4);
cout << endl;
}
```

---

### a795binf.cpp

```
// nuovi elementi: ios::binary, clear()
// nuovi concetti: file binario
#include <fstream.h>
#include <iomanip.h>
#include <stdlib.h>

void acapo() {
while (cin.get() != '\n') continue;
}

struct tfilm {
    char titolo[36];    // titolo del film
    char regista[22];   // nome del regista
    int anno;           // anno di uscita
    float valutazione;  // valutazione del film
};

const char * const file = "films.dat";

int main()
{
cout.setf(ios::showpoint);
cout.setf(ios::fixed, ios::floatfield);
cout << setprecision(1);

tfilm film;

// mostra il contenuto iniziale
ifstream fin;
fin.open(file, ios::in | ios::binary);   // file binario
if (fin) {
   cout << "Il contenuto del file "
        << file << " e' il seguente:\n";
   while (fin.read((char *) &film, sizeof film)) {
       cout.setf(ios::left, ios::adjustfield);
       cout << setw(30) << film.titolo << ": "
            << setw(20) << film.regista;
       cout.setf(ios::right, ios::adjustfield);
       cout << setw(8)  << film.anno;
       cout << setw(8)  << film.valutazione << "\n";
       }
```

```
   }
fin.close();

// aggiungi altri dati
ofstream fout(file, ios::out | ios::app | ios::binary);
if (!fout) {
    cerr << "Non riesco ad aprire il file " << file << " in scrittura!\n";
    exit(1);
    }

cout << "Introdurre il titolo del film (<Invio> per terminare):\n";
cin.get(film.titolo, 36);
while (film.titolo[0] != '\0') {
    acapo();
    cout << "Regista: ";
    cin.get(film.regista,22);
    acapo();
    cout << "Anno: ";
    cin >> film.anno;
    cout << "Valutazione (0-10): ";
    cin >> film.valutazione;
    fout.write((char *) &film, sizeof film);
    acapo();
    cout << "Introdurre il titolo del film"
            " (<Invio> per terminare):\n";
    cin.get(film.titolo, 36);
    }
fout.close();

// mostra il file modificato
fin.clear();   // non necessario in Turbo C++
fin.open(file, ios::in | ios::binary);
if (fin) {
   cout << "Il seguente e' il contenuto del file " << file
        << " dopo l'aggiornamento:\n";
   while (fin.read((char *) &film, sizeof film)) {
       cout.setf(ios::left, ios::adjustfield);
       cout << setw(30) << film.titolo << ": "
            << setw(20) << film.regista;
       cout.setf(ios::right, ios::adjustfield);
       cout << setw(8)  << film.anno
            << setw(8)  << film.valutazione << "\n";
       }
   }
fin.close();

return 0;
}
```

---

### a800rand.cpp

```
// nuovi elementi: seekg(), eof(), streampos, fail()
// nuovi concetti: accesso random su file binario
#include <fstream.h>
#include <iomanip.h>
#include <stdlib.h>
#include <string.h>

void acapo() {
while (cin.get() != '\n') continue;
}

struct tfilm {
    char titolo[36];    // titolo del film
    char regista[22];   // nome del regista
    int anno;           // anno di uscita
```

```
    float valutazione;  // valutazione del film
};

const char * file = "films.dat";

int main()
{
tfilm film;
cout.setf(ios::showpoint);
cout.setf(ios::fixed, ios::floatfield);
cout << setprecision(1);

// mostra il contenuto iniziale
fstream fin;     // stream di lettura-scrittura
fin.open(file, ios::in | ios::out | ios::binary);
int ct = 0;
if (fin) {
   fin.seekg(0);    // vai all'inizio del file
   cout << "Il contenuto del file "
        << file << " e' il seguente:\n";
   while (fin.read((char *) &film, sizeof film)) {
       cout << setw(2) << ct++ << ": ";
       cout.setf(ios::left, ios::adjustfield);
       cout << setw(30) << film.titolo << ": "
            << setw(20) << film.regista;
       cout.setf(ios::right, ios::adjustfield);
       cout << setw(8)  << film.anno;
       cout << setw(8)  << film.valutazione << "\n";
       }
    if (fin.eof())
        fin.clear(); // azzera flag eof
    else {
        cerr << "Errore durante la lettura di " << file << ".\n";
        exit(1);
        }
    }
else {
   cerr << "Non e' possibile aprire il file " << file << "\n";
   exit(2);
   }

// modifica un record
cout << "Introdurre il numero del record da modificare: ";
long rec;
cin >> rec;
acapo();              // elimina il newline
if (rec < 0 || rec >= ct) {
   cerr << "Numero di record non valido!\n";
   exit(3);
   }
streampos place = rec * sizeof film;  // converti nel tipo streampos (?)
fin.seekg(place);    // accesso random
if (fin.fail()) {
   cerr << "Errore nel tentativo di posizionamento\n";
   exit(4);
   }

fin.read((char *) &film, sizeof film);
cout << "Record selezionato:\n";
cout << rec << ": ";
cout.setf(ios::left, ios::adjustfield);
cout << setw(30) << film.titolo << ": "
     << setw(20) << film.regista;
cout.setf(ios::right, ios::adjustfield);
cout << setw(8)  << film.anno;
cout << setw(8)  << film.valutazione << "\n";
if (fin.eof())
    fin.clear();     // azzera flag eof

cout << "Introdurre il titolo del film: ";
```

```
char riga[80];
cin.get(riga, 36);
acapo();
if (riga[0]) strncpy(film.titolo,riga,36);
cout << "Regista: ";
cin.get(riga, 22);
acapo();
if (riga[0]) strncpy(film.regista,riga,22);
cout << "Anno: ";
if (cin.peek() != '\n') cin >> film.anno;
acapo();
cout << "Valutazione (0-10): ";
if (cin.peek() != '\n') cin >> film.valutazione;
acapo();
fin.seekp(place);    // torna indietro
fin.write((char *) &film, sizeof film);

if (fin.fail()) {
   cerr << "Errore nel tentativo di scrittura\n";
   exit(5);
   }

// mostra il file modificato
ct = 0;
fin.seekg(0);            // torna all'inizio del file

cout << "Il seguente e' il contenuto del file " << file
     << " dopo l'aggiornamento:\n";
while (fin.read((char *) &film, sizeof film)) {
    cout << setw(2) << ct++ << ": ";
    cout.setf(ios::left, ios::adjustfield);
    cout << setw(30) << film.titolo << ": "
         << setw(20) << film.regista;
    cout.setf(ios::right, ios::adjustfield);
    cout << setw(8)  << film.anno
         << setw(8)  << film.valutazione << "\n";
    }

fin.close();

return 0;
}
```

---

## a805dump.cpp

```
// nuovi elementi: fill(), isgraph()
#include <fstream.h>
#include <conio.h>
#include <stdlib.h>
#include <iomanip.h>
#include <ctype.h>

/* dump esadecimale e ASCII di un file */

int main(int argc, char * argv[])
{
if (argc == 1)             // termina se non ci sono argumenti
   {
   cerr << "Uso: " << argv[0] << " <nomefile>\n";
   exit(1);
   }
unsigned char b, c=0, r=0;
unsigned int ind=0;
char riga[17] = "                ";  // 16 spazi
clrscr();
ifstream fin(argv[1]);
if (!fin)
```

```
   {
   cerr << "Non sono riuscito ad aprire il file " << argv[1] << "\n";
   exit(2);
   }
cout.fill('0');
cout << "  " << hex << setw(4) << ind << "  ";

while (fin.get(b)) {
  ind++;
  cout << " " << setw(2) << int(b);
  riga[c] = isgraph(b) ? b : ' ';
  if (++c==8) cout << " ";
  else if (c==16) {
    cout << "   " << riga << endl;
    c=0;
    if (++r==16) {
      getch();
      cout << endl;
      r=0;
      }
    if (fin.peek() != EOF)
      cout << "  " << setw(4) << ind << "  ";
    }
  }

fin.close();

if (c<16) {
  for (r=c; r < 16; r++) {
    cout << "   ";
    riga[r] = ' ';
    }
  if (c<8) cout << " ";
  cout << "   " << riga << endl;
  }
getch();
return 0;
}
```

---

### a810bgar.cpp

```
typedef int array30K[15000];
array30K a,b,c;    // 90000 byte (errore! supera 64K)
void main() {
}
```

---

### a815hpar.cpp

```
#include <iostream.h>

int *a, *b, *c;    // solo 3 puntatori
void main() {
a = new int[15000];
b = new int[15000];
c = new int[15000];
a[1] = 12;
a[14999] = -44;
*(b+2) = 55;
cout << b[2] << endl;
}
```

## a820dstr.cpp

```
#include <fstream.h>
#include <stdlib.h>
#include <conio.h>
#include <string.h>
//#include <alloc.h>

void main(int argc, char * argv[]) {
if (argc == 1)             // termina se non ci sono argumenti
   {
   cerr << "Uso: " << argv[0] << " <nomefile>\n";
   exit(1);
   }
ifstream fin(argv[1]);
if (!fin)
   {
   cerr << "Non sono riuscito ad aprire il file " << argv[1] << "\n";
   exit(2);
   }
char riga[80];
char * righe[1000];
int n;
clrscr();
//cout << coreleft() << endl;
for (n=0; fin.getline(riga,80); n++) {
   righe[n] = new char[strlen(riga)+1];
   strcpy(righe[n],riga);
   }
fin.close();
//cout << coreleft() << endl;
for (n--;n>=0;n--) {
   cout << righe[n] << endl;
   delete [] righe[n];
   }
//cout << coreleft() << endl;
}
```

## a825invl.cpp

```
// nuovi elementi: operatore freccia ( -> )
// nuovi concetti: lista dinamica
#include <iostream.h>

struct Lnodo {
       char dato;
       Lnodo *succ;
       };

void main () {
Lnodo *base=NULL, *p;
cout << "Introdurre una sequenza di caratteri (^Z per terminare):" << endl;
char ch;
while (cin.get(ch)) {
   p = new Lnodo;
   p->dato = ch;    // p->dato  equivale a  (*p).dato
   p->succ = base;
   base = p;
   }
while (p) {
   cout << p->dato;
   base = p;
   p = p->succ;
   delete base;
```

```
   }
}
```

---

### a830invl.cpp

```
#include <iostream.h>

struct Lnodo {
       char dato;
       Lnodo *succ;
       };

void MostraLista(Lnodo*);


void main () {
Lnodo *base=NULL, *p;
cout << "Introdurre una sequenza di caratteri (^Z per terminare):" << endl;
char ch;
while (cin.get(ch)) {
   p = new Lnodo;
   p->dato = ch;
   p->succ = base;
   base = p;
   }
MostraLista(base);
}


void MostraLista(Lnodo* lista) {
while (lista) {
   cout << lista->dato;
   lista = lista->succ;
   }
}
```

---

### a835invl.cpp

```
// nuovi concetti: liste e ricorsione
#include <iostream.h>

struct Lnodo {
       char dato;
       Lnodo *succ;
       };

void MostraLista(Lnodo*);


void main () {
Lnodo *base=NULL, *p;
cout << "Introdurre una sequenza di caratteri (^Z per terminare):" << endl;
char ch;
while (cin.get(ch)) {
   p = new Lnodo;
   p->dato = ch;
   p->succ = base;
   base = p;
   }
MostraLista(base);
}
```

```
void MostraLista(Lnodo* lista) {
if (lista) {
   cout << lista->dato;
   MostraLista(lista->succ);
   }
}
```

---

**a840copl.cpp**

```
#include <iostream.h>
//#include <alloc.h>

struct Lnodo {
       char dato;
       Lnodo *succ;
       };

void AggiungiNodo(Lnodo*&);
void MostraLista(Lnodo*);
void DistruggiLista(Lnodo*&);


void main () {
Lnodo *base, *p;
cout << "Introdurre una sequenza di caratteri (^Z per terminare):" << endl;
base = NULL;
//cout << coreleft() << endl;
while (cin.peek() != EOF)
   AggiungiNodo(base);
MostraLista(base);
//cout << coreleft() << endl;
DistruggiLista(base);
//cout << coreleft() << endl;
}

void AggiungiNodo(Lnodo*& lista) {
if (lista)
   AggiungiNodo(lista->succ);
else {
   lista = new Lnodo;
   lista->succ = NULL;
   cin.get(lista->dato);
   }
}


void MostraLista(Lnodo* lista) {
if (lista) {
   cout << lista->dato;
   MostraLista(lista->succ);
   }
}

void DistruggiLista(Lnodo*& testa) {
if (testa) {
   DistruggiLista(testa->succ);
   delete testa;
   }
}
```

---

**a845list.cpp**

```
// nuovi concetti: manipolazione di liste
//                 parametro stream
#include <fstream.h>
#include <alloc.h>

struct Lnodo {
       int dato;
       Lnodo *succ;
       };

void writelist(Lnodo*);
void InserisciPrima(Lnodo*&, Lnodo*, Lnodo*);
void InserisciDopo(Lnodo*&, Lnodo*, Lnodo*);
void InvertiLista(Lnodo*&);
int  ListeUguali(Lnodo*, Lnodo*);
void LeggiInTesta(ifstream&, Lnodo*&);
void LeggiInCoda(ifstream&, Lnodo*&);
void DistruggiLista(Lnodo*&);


void main () {
Lnodo *lista1=NULL, *lista2=NULL;
ifstream fin("lista.txt");
cout << "\nDati letti dal file:\n";
cout << coreleft() << endl;
LeggiInTesta(fin,lista1);
cout << coreleft() << endl;
cout << "Dati come inseriti nella lista N. 1:\n";
writelist(lista1);
fin.clear();
fin.seekg(0);
cout << "\nDati letti dal file:\n";
cout << coreleft() << endl;
LeggiInCoda(fin,lista2);
cout << coreleft() << endl;
fin.close();
cout << "Dati come inseriti nella lista N. 2:\n";
writelist(lista2);
InvertiLista(lista2);
cout << "\nContenuto della lista N. 2 rovesciata:" << endl;
writelist(lista2);
if (ListeUguali(lista1,lista2))
   cout << "OK";
else
   cout << "Oh Oh...!!!";
cout << endl;
cout << coreleft() << endl;
DistruggiLista(lista2);
cout << coreleft() << endl;
DistruggiLista(lista1);
cout << coreleft() << endl;
}


void writelist(Lnodo* ptr) {
while (ptr) {
   cout << ptr->dato << " ";
   ptr = ptr->succ;
   }
cout << endl;
}


void InserisciPrima(Lnodo* &testa, Lnodo* posto, Lnodo* nuovo) {
// Inserisce il nodo 'nuovo' nella lista 'testa' prima di 'posto'
if (testa == NULL) {
   nuovo->succ = NULL;
   testa = nuovo;
   }
else if (posto==testa) {
```

```
   nuovo->succ = testa;
   testa = nuovo;
   }
else {
   Lnodo* precedente;
   for(precedente=testa;precedente->succ!=posto;precedente=precedente->succ)
      continue;
   nuovo->succ = posto;
   precedente->succ = nuovo;
   }
}


void InserisciDopo(Lnodo* &testa, Lnodo* posto, Lnodo* nuovo) {
// Inserisce il nodo 'nuovo' nella lista 'testa' dopo 'posto'
if (testa == NULL) {
   nuovo->succ = NULL;
   testa = nuovo;
   }
else {
   nuovo->succ = posto->succ;
   posto->succ = nuovo;
   }
}


void InvertiLista(Lnodo*& testa) {
// Inverte la lista 'testa' (il nodo di coda diventa il nodo di testa e viceversa)
Lnodo *prima=NULL, *corrente=testa, *dopo;
while (corrente) {
   dopo = corrente->succ;
   corrente->succ = prima;
   prima = corrente;
   corrente = dopo;
   }
testa = prima;
}


int ListeUguali(Lnodo* l1, Lnodo* l2) {
// Restituisce vero (1) se le due liste sono uguali, cioe'
// se le due liste sono entrambe vuote, oppure
// il contenuto della testa coincide e le liste private della testa
// sono uguali
return (!(l1 || l2) ||
        l1 && l2 && l1->dato==l2->dato &&
        ListeUguali(l1->succ,l2->succ));
}


void LeggiInTesta(ifstream& fin, Lnodo*& testa) {
int n;
Lnodo *ptr;
testa = NULL;
while (fin >> n) {
   ptr = new Lnodo;
   ptr->dato = n;
   cout << n << " ";
   InserisciPrima(testa, testa, ptr);
   }
cout << endl;
}


void LeggiInCoda(ifstream& fin, Lnodo*& testa) {
int n;
Lnodo *ptr, *ultimo;
testa = NULL;
while (fin >> n) {
   ptr = new Lnodo;
```

```
   ptr->dato = n;
   cout << n << " ";
   InserisciDopo(testa, ultimo, ptr);
   ultimo = ptr;
   }
cout << endl;
}

void DistruggiLista(Lnodo*& testa) {
while (testa) {
   Lnodo* ptr = testa;
   testa = testa->succ;
   delete ptr;
   }
}
```

---

### a850albe.cpp

```
#include <fstream.h>

typedef int Tipo;

struct Tnodo
   {
   Tipo info;
   Tnodo *sin, *des;
   };

void inserisci(Tnodo*&, Tnodo*);
void InOrdine(Tnodo*);
void EliminaAlbero(Tnodo*&);


main () {
Tnodo *albero=NULL, *ptr;
ifstream fin("lista.txt");
cout << "\nDati letti dal file:\n";
Tipo n;
while (fin >> n)
   {
   cout << n << " ";
   ptr = new Tnodo;
   ptr->info = n;
   ptr->sin = ptr->des = NULL;
   inserisci(albero,ptr);
   }
fin.close();
cout << "\nVisita InOrdine:\n";
InOrdine(albero);
EliminaAlbero(albero);
return 0;
}


void inserisci(Tnodo*& radice, Tnodo* elemento)
{
if (radice)
   if (radice->info >= elemento->info)
      inserisci(radice->sin,elemento);
   else
      inserisci(radice->des,elemento);
else
   radice = elemento;
}
```

```
void InOrdine(Tnodo* radice)
{
if (radice)
   {
   InOrdine(radice->sin);
   cout << radice->info << endl;
   InOrdine(radice->des);
   }
}


void EliminaAlbero(Tnodo*& radice) {
if (radice) {
   EliminaAlbero(radice->sin);
   EliminaAlbero(radice->des);
   delete radice;
   }
}
```

---

**a860fraz.cpp**

```
#include <iostream.h>

struct Razionale {
   int num, den;
};

void assign(Razionale&, int, int);
double convert(Razionale);
void invert(Razionale&);
void print(Razionale);


int main() {
Razionale x;
assign(x,22,7);
cout << "x = " ;  print(x);
cout << " = " << convert(x) << endl;
invert(x);
cout << "1/x = ";  print(x);  cout << endl;
return 0;
}

void assign(Razionale& f, int numerator, int denominator) {
f.num = numerator;
f.den = denominator;
}

double convert(Razionale f) {
return double(f.num)/f.den;
}

void invert(Razionale& f) {
int temp = f.num;
f.num = f.den;
f.den = temp;
}

void print(Razionale f) {
cout << f.num << '/' << f.den;
}
```