

Principali esempi di codice sorgente

Capitolo 2 - Salve, mondo

```
#include <stdio.h>
/* Il mio primo programma C */
main()
{
    printf("salve, mondo");
}
```

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    printf("salve, mondo");
    system("PAUSE");
    return 0;
}
```

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

int main()
{
    printf("Hello world!\n");
    return 0;
}
```

Capitolo 3 - Le variabili ed il mondo esterno

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

int main(int argc, char *argv[])
{
    int x;
    x = 3;
    system("PAUSE");
    return 0;
}
```

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

int main(int argc, char *argv[])
{
    int x;
    x = 3;
    printf("Il valore della variabile e': %d", x);
    system("PAUSE");
    return 0;
}
```

```

}

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

int main(int argc, char *argv[])
{
    int x;

    scanf("%d", &x);
    printf("Il numero digitato e': %d \n", x);

    system("PAUSE");
    return 0;
}

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

int main(int argc, char *argv[])
{
    printf(" - Le variabili del C: occupazione di memoria in byte - \n\n");
    printf("char:\t%d\n", sizeof(char));
    printf("short:\t%d\n", sizeof(short));
    printf("int:\t%d \n", sizeof(int));
    printf("long:\t%d \n", sizeof(long));
    printf("float:\t%d \n", sizeof(float));
    printf("double:\t%d \n", sizeof(double));
    printf("\n");

    system("PAUSE");
    return 0;
}

#define PI_GRECO 3.14
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    float raggio;
    float area;
    printf("Inserisci il raggio del cerchio: ");
    scanf ("%f", &raggio);
    area=raggio*raggio*PI_GRECO;
    printf("L'area del cerchio e': %f\n", area);
    system("PAUSE");
    return 0;
}

```

Capitolo 4 - Operiamo sui dati

```

/* calcola il precedente e successivo di un numero inserito da tastiera */
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int x, prec, succ;
    printf("Dammi un numero: ");
    scanf("%d", &x);

    prec=x-1;
    printf("Il numero precedente e': %d\n", prec);
}

```

```

succ=x+1;
printf("Il numero successivo e': %d\n", succ);

system("PAUSE");
return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    //sezione dichiarativa
    int x, prec, succ;

    //sezione input
    printf("Dammi un numero: ");
    scanf("%d", &x);

    //elaborazione
    prec=x-1;
    succ=x+1;

    //sezione output
    printf("Il numero precedente e': %d\n", prec);
    printf("Il numero successivo e': %d\n", succ);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    const float pi = 3.14159;
    printf("%10.2f\n", pi);
    printf("%010.2f\n", pi);
    printf("%+10.2f\n", pi);

    system("PAUSE");
    return 0;
}

```

Capitolo 5 - Come fare le pulci ad un programma: il debugging di un'applicazione

```

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

```

```

int main(int argc, char *argv[])
{
    int x;
    int y;
    int s;

    x = 3;
    y = 4;

    s = x + y;

    printf ("La somma e': %d\n", s);
}

```

```

    system("PAUSE");
    return 0;
}

```

Capitolo 6 - Istruzioni condizionali: operiamo delle scelte

```

/* Determina il maggiore tra due interi, input utente */
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int a, b;

    printf("Inserisci il primo numero: ");
    scanf("%d", &a);
    printf("Inserisci il secondo numero: ");
    scanf("%d", &b);
    if (a > b)
    {
        printf("Il primo numero e' maggiore del secondo");
    }
    else
    {
        printf("Il secondo numero e' maggiore del primo");
    }

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int voto;
    printf ("Inserisci il tuo voto: ");
    scanf("%d", &voto);
    if (voto < 1 || voto > 10)
    {
        printf("Devi inserire un voto compreso tra 1 e 10");
    }
    else
    {
        if (voto < 6)
        {
            printf("Insufficiente");
        }
        if (voto == 6)
        {
            printf("Sufficiente");
        }
        if (voto == 7)
        {
            printf("Discreto");
        }
        if (voto == 8)
        {
            printf("Buono");
        }
        if (voto == 9)
        {
            printf("Ottimo");
        }
    }
}

```

```

        if (voto == 10)
        {
            printf("Eccellente");
        }
    }

    printf("\n");

    system("PAUSE");
    return 0;
}

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

int main(int argc, char *argv[])
{
    int voto;
    printf ("Inserisci il tuo voto: ");
    scanf("%d", &voto);
    switch(voto)
    {
        case 1:
        case 2:
        case 3:
        case 4:
        case 5:
            printf("Insufficiente");
            break;
        case 6:
            printf("Sufficiente");
            break;
        case 7:
            printf("Discreto");
            break;
        case 8:
            printf("Buono");
            break;
        case 9:
            printf("Ottimo");
            break;
        case 10:
            printf("Eccellente");
            break;
        default:
            printf("Devi inserire un voto compreso tra 1 e 10");
            break;
    }

    system("PAUSE");
    return 0;
}

```

Capitolo 7 – Cicli: noi la mente, la macchina il braccio

```

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

int main(int argc, char *argv[])
{
    int i;
    for (i=1; i<=100; i++)

```

```

    {
        printf("Carlo Mazzone\n");
    }
    system("PAUSE");
    return 0;
}

```

```

/* Stampa numeri pari fino a 50 */
#include <stdio.h>
#include <stdlib.h>

```

```

int main(int argc, char *argv[])
{
    int i;
    for (i=1; i<=50; i++)
    {
        if (i%2 == 0)
        {
            printf("%2d\n", i);
        }
    }

    system("PAUSE");
    return 0;
}

```

```

/* Stampa numeri pari fino a 50 - versione senza if */
#include <stdio.h>
#include <stdlib.h>

```

```

int main(int argc, char *argv[])
{
    int i;
    for (i=2; i<=50; i=i+2)
    {
        printf("%2d\n", i);
    }

    system("PAUSE");
    return 0;
}

```

```

#include <stdio.h>
#include <stdlib.h>
#define TOT_NUM 10

```

```

int main(int argc, char *argv[])
{
    int max, i, x;

    for (i=1; i<= TOT_NUM; i++)
    {
        printf("Dammi il numero: ");
        scanf("%d", &x);
        if (i==1) max = x; /* impostiamo il primo numero letto come max temporaneo */
        if (x>max)
        {
            max=x;
        }
    }

    printf("Il massimo e': %d\n", max);

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int i;
    i=1;
    while (i<=10)
    {
        printf("Io sono la riga numero %d\n", i);
        i++;
    }

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int i, x, s;
    i=1;
    s=0;
    while(i <= 5)
    {
        printf("Inserisci un numero: ");
        scanf("%d", &x);
        s = s + x;

        i++;
    }
    printf("La somma e': %d\n", s);
    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int i, x, s;
    s=0;
    for (i=1; i <= 5; i++)
    {
        printf("Inserisci un numero: ");
        scanf("%d", &x);
        s = s + x;
    }

    printf("La somma e': %d\n", s);
    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int x, s;
    s = 0;
    printf("Inserisci un numero positivo oppure 0 per terminare: ");
    scanf("%d", &x);
}

```

```

while (x != 0)
{
    s = s + x;
    printf("Inserisci un numero positivo oppure 0 per terminare: ");
    scanf("%d", &x);
}

printf("La somma e': %d\n", s);
system("PAUSE");
return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    int x, s;
    s = 0;

    do
    {
        printf("Inserisci un numero positivo oppure 0 per terminare: ");
        scanf("%d", &x);
        s = s + x;
    }while (x != 0);

    printf("La somma e': %d\n", s);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    int count=0;
    int x;
    printf("Digita un numero per volta confermandolo con Invio - 0 per terminare\n");
    do
    {
        scanf("%d", &x);
        count++;
    }while (x!=0); /* continuo il ciclo finché non trovo il carattere zero */

    /* decremento di 1 il valore di count per escludere lo zero dal conteggio */
    printf("Lunghezza sequenza: %d\n ", count-1);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    int i, x;
    printf("Da 1 a 100 dimmi quando ti vuoi fermare: ");
    scanf("%d", &x);
    i = 1;
    while (i<=100)
    {
        if (i == x)
        {

```



```

        break;
    }
    printf("%d\n", i);
    i++;
}

system("PAUSE");
return 0;
}

```

Capitolo 8 - Cominciamo a fare sul serio: gli array

```

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

int main(int argc, char *argv[])
{
    /* Dichiarazione array */
    int pippo[5];

    /* Valorizzazione degli elementi */
    pippo[0]=2;
    pippo[1]=4;
    pippo[2]=6;
    pippo[3]=8;
    pippo[4]=10;

    /* Stampa array */
    printf("Il contenuto dell'array e':");
    printf("\n%d", pippo[0]);
    printf("\n%d", pippo[1]);
    printf("\n%d", pippo[2]);
    printf("\n%d", pippo[3]);
    printf("\n%d", pippo[4]);
    printf("\n");

    system("PAUSE");
    return 0;
}

```

```

#include <stdio.h>
#include <stdlib.h>
#define TOT_NUM 5

int main(int argc, char *argv[])
{
    int i;
    int a[TOT_NUM];

    /* Leggo i numeri in input */
    printf("Inserisci i singoli numeri confermandoli con il tasto Invio\n");
    for (i = 0; i < TOT_NUM; i++)
    {
        scanf("%d", &a[i]);
    }

    /* Stampo i numeri inseriti */
    printf("I numeri inseriti sono:\n");
    for (i = 0; i < TOT_NUM; i++)
    {
        printf("%d%c", a[i], ' ');
    }

    printf("\n");
}

```

```
    system("PAUSE");
    return 0;
}
```

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

```
int main(int argc, char *argv[])
{
    int a[5];
    int i;

    printf("I valori sono:\n");
    for (i = 0; i < 5; i++)
    {
        printf("%d\n", a[i]);
    }

    printf("\n");

    system("PAUSE");
    return 0;
}
```

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

```
int main(int argc, char *argv[])
{
    char c;
    c = 'A';
    printf ("Il valore numerico della carattere %c e': %d \n", c, c);

    system("PAUSE");
    return 0;
}
```

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

```
int main(int argc, char *argv[])
{
    int i;
    for(i=0; i<=255; i++)
    {
        printf ("Il valore numerico del simbolo %c e': %d \n", i, i);
    }

    system("PAUSE");
    return 0;
}
```

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

```
int main(int argc, char *argv[])
{
    char c;
    printf ("Premi un tasto: ");
    c = getchar();
    printf ("Il valore numerico del simbolo %c e': %d \n", c, c);

    system("PAUSE");
    return 0;
}
```

```

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

int main(int argc, char *argv[])
{
    char c;
    printf("Inserisci un carattere: ");
    c = getchar ();
    printf("Il carattere inserito e': ");
    putchar(c);

    system("pause");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    char nome[20];
    printf("Come ti chiami: ");
    scanf("%s", &nome);
    printf("Il tuo nome e' %s \n", nome);

    system("pause");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    char nome[20];
    printf("Come ti chiami? ");
    gets(nome);
    printf("Il tuo nome e': %s \n", nome);

    system("pause");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int i=0;
    char nome[20];
    char c;
    printf("Come ti chiami? ");

    while ((c = getchar()) != '\n')
    {
        nome[i] = c;
        i++;
    }
    nome[i] = '\0';

    printf("Il tuo nome e': %s\n" , nome);

    system("pause");
    return 0;
}

```

```

}

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

int main(int argc, char *argv[])
{
    int len = 0;
    char nome[] = "Ciccio Pasticcio";
    len = strlen(nome);
    printf("Il tuo nome e' lungo %d caratteri\n", len);

    system("pause");
    return 0;
}

```

```

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

main()
{
    char nome1[] = "Ciccio Pasticcio";
    char nome2[] = "Pluto De Plutis";
    strcpy(nome1, nome2);
    printf("Il tuo nome e' %s ", nome1);

    system("pause");
    return 0;
}

```

```

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

main()
{
    char nome[20];
    strcpy(nome, "Carlo A.");
    printf("Il tuo nome e' %s ", nome);

    system("pause");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    char nome1[] = "Ciccio";
    char nome2[] = " Pasticcio";

    strcat(nome1, nome2);
    printf("Il tuo nome e' %s ", nome1);

    system("pause");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{

```

```

int risultato = -1;
char password[] = "mago merlino";
char tentativo[20];

printf("Digita la password: ");
gets(tentativo);
risultato = strcmp(password, tentativo);

if (risultato == 0)
{
    printf("Accesso consentito");
}
else
{
    printf("ACCESSO NEGATO");
}

system("pause");
return 0;
}

```

Capitolo 9 - Array bidimensionali

```

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

int main(int argc, char *argv[])
{
    int tabella[4][3]={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
    int i;
    for (i=0; i<=2; i++)
    {
        printf("%d ", tabella[0][i]);
    }

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    int tabella[4][3]={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
    int i;

    /* stampa prima riga */
    for (i=0; i<=2; i++)
    {
        printf("%d ", tabella[0][i]);
    }
    printf("\n");

    /* stampa seconda riga */
    for (i=0; i<=2; i++)
    {
        printf("%d ", tabella[1][i]);
    }
    printf("\n");

    /* stampa terza riga */
    for (i=0; i<=2; i++)
    {
        printf("%d ", tabella[2][i]);
    }
}

```

```

printf("\n");

/* stampa quarta riga */
for (i=0; i<=2; i++)
{
    printf("%d ", tabella[3][i]);
}
printf("\n");

system("PAUSE");
return 0;
}

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

int main(int argc, char *argv[])
{

    int tabella[4][3]={{1,2,3},{4,5,6},{7,8,9},{10,11,12}};
    int i,j;

    for (i=0; i<4; i++)
    {
        for (j=0; j<3; j++)
        {
            printf("%3d", tabella[i][j]);
        }
        printf("\n");
    }

    system("PAUSE");
    return 0;
}

#include <time.h> /* necessario per la funzione time */
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int x;
    srand(time(NULL));
    x = rand();
    printf("Numero casuale: %d", x);

    system("PAUSE");
    return 0;
}

#include <time.h> /* necessario per la funzione time */
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int pos; /* Posizione casuale vincente */
    int x; /* Puntata del giocatore */
    srand(time(NULL));
    pos = rand()%3+1;
    printf("IL GIOCO DELLE TRE CARTE\n\n");
    printf("In quale posizione si trova la carta vincente?\n");
    printf("Indica un numero da 1 a 3 e buona fortuna!\n");
    scanf("%d", &x);
    if (x == pos)
    {
        printf("Complimenti hai vinto!!!\n");
    }
    else
    {
        printf("Peccato, hai perso! La posizione vincente era la numero %d\n", pos);
    }
}

```

```

    }
    system("PAUSE");
    return 0;
}

#include <time.h> /* necessario per la funzione time */
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    int pos; /* Posizione casuale vincente */
    int x; /* Puntata del giocatore */
    srand(time(NULL));

    while(x!=0)
    {
        pos = rand()%3+1;
        printf("IL GIOCO DELLE TRE CARTE\n\n");
        printf("In quale posizione si trova la carta vincente?\n");
        printf("Indica un numero da 1 a 3 e buona fortuna! Digita 0 per uscire dal gioco.\n");
        scanf("%d", &x);
        if (x==0)
        {
            break;
        }
        else if (x==pos)
        {
            printf("Complimenti hai vinto!!!\n");
        }
        else
        {
            printf("Peccato, hai perso! La posizione vincente era la numero %d\n", pos);
        }
    }

    printf("Ciao ed alla prossima partita!!!\n");
    system("PAUSE");
    return 0;
}

```

Capitolo 10 – Ordiniamo le cose: davvero “notevole”

```

#include <stdio.h>
#include <stdlib.h>
#define N 5

int main(int argc, char *argv[])
{
    int a[N] = {9, 5, 1, 3, 2};
    int i, j, temp;

    for (i=0; i <= N - 2; i++)
    {
        for (j=i+1; j <= N -1; j++ )
        {
            if (a[i] > a[j])
            {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
            } /* fine if */
        } /* fine for j */
    }
}

```

```

} /* fine for i */

/* stampo array finale */
for (i=0; i < N; i++)
{
    printf ("%3d", a[i]);
}

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

#include <stdio.h>
#include <stdlib.h>
#define N 5

int main(int argc, char *argv[])
{
    int a[N]={3, 4, 1, 5, 2};

    int i, j, temp;

    for(i=1; i<N; i++)
    {
        for(j=0; j<N-i; j++)
        {
            if(a[j]>a[j+1])
            {
                temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            } /* fine if */
        } /* fine for j */
    } /* fine for i */

    for(i=0; i<N; i++)
    {
        printf("%d ",a[i]);
    }
    printf("\n");

    system("PAUSE");
    return 0;
}

```

Capitolo 11 – Soluzioni più semplici per problemi più complessi: le funzioni

```

#include <stdio.h>
#include <stdlib.h>
void banner (void); /* dichiarazione funzione */
int main(int argc, char *argv[])
{
    banner(); /* qui richiamo il banner */

    system("PAUSE");
    return 0;
}

void banner (void)
{
    printf(" ##### #          ### \n");
    printf(" #      #  ## ##### #      ##### ##      # # \n");
    printf(" #          # # # # # # # # # # # # # # \n");
    printf(" #          #  ## # # # # # # # # # # # \n");
}

```



```

printf(" #          ##### ##### #      #      #      #      #      # \n");
printf(" #          # #      # #      # #      # #      #      #      # \n");
printf(" ##### #      # #      # ##### #####      ##### # #      # #      # \n");
}

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

void n_linee (int);

int main(int argc, char *argv[])
{
    /* inserisco 5 linee */
    n_linee(5);

    /* inserisco 2 linee */
    n_linee(2);

    /* inserisco 5 linee */
    n_linee(5);

    system("PAUSE");
    return 0;
}

void n_linee (int n)
{
    int i;
    for (i=1; i <= n; i++)
    {
        printf("\n");
    }
}

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

/* Dichiarazione della funzione potenza
che prende in input due interi e restituisce un intero */
int potenza(int, int);

int main(int argc, char *argv[])
{
    int b, e;
    printf("Inserisci la base: ");
    scanf("%d", &b);
    printf("Inserisci l'esponente: ");
    scanf("%d", &e);

    printf("La potenza con base %d ed esponente %d vale %d \n", b, e, potenza(b, e));

    system("PAUSE");
    return 0;
}

/* implementazione funzione potenza */
int potenza(int x, int n)
{
    int i;
    int p = 1;

    for (i=1; i<=n; i++)
    {
        p = p*x;
    }
    return p;
}

```

```

/* CALCOLA LA SERIE DI POTENZE CON BASE 2 AVENTI COME ESPONENTE
I PRIMI N NUMERI NATURALI */

#include <stdio.h>
#include <stdlib.h>
/* dichiarazione della funzione potenza
che prende in input due interi e restituisce un intero */

int potenza(int, int);

int main(int argc, char *argv[])
{
    int n, i;
    printf("\nInserisci il numero N: ");
    scanf("%d", &n);
    if (n < 0)
    {
        printf("\nIl numero non può essere negativo!");
        return;
    }

    for (i=1; i<=n;i++)
    {
        printf("Potenza base %d esponente %d = %d \n", 2, i, potenza(2, i));
    }

    system("PAUSE");
    return 0;
}

/* implementazione funzione potenza */
int potenza(int x, int n)
{
    int i;
    int p = 1;

    for (i=1; i<=n; i++)
    {
        p = p*x;
    }
    return p;
}

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

int var = 0;
void incrementa(void);

int main(int argc, char *argv[])
{
    printf("Prima della chiamata: %d\n", var);
    incrementa();
    printf("Dopo la chiamata: %d\n", var);
    system("PAUSE");
    return 0;
}

void incrementa(void)
{
    extern int var;
    var = var + 1;
    return;
}

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

int tictac(void);
int main(int argc, char *argv[])

```

```

{
    int i;
    printf("Il mio contatore\n\n");
    for (i=1; i<=10; i++)
    {
        printf("%d\n", tictac());
    }
    printf("\n");

    system("PAUSE");
    return 0;
}

int tictac(void)
{
    int x = 0;
    x = x + 1;

    return x;
}

```

Capitolo 12 - La ricorsione e le funzioni ricorsive

```

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

long fattoriale(long);

int main(int argc, char *argv[])
{
    int n;
    printf("Calcolo del fattoriale - Digita un numero naturale: ");
    scanf("%d", &n);
    printf("Il fattoriale del numero %d e': %ld\n", n, fattoriale(n));

    system("PAUSE");
    return 0;
}

long fattoriale(long x)
{
    if (x==0)
        return 1;
    else
        return x*fattoriale(x-1);
}

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

int main()
{
    int n, i;
    long int fatt = 1;

    printf("Inserisci un numero : ");
    scanf("%d",&n);

    for (i = 1; i <= n; i++)
    {
        fatt = fatt * i;
    }

    printf("Il fattoriale di %d e' %ld\n", n, fatt);

    system("PAUSE");
}

```

```

    return 0;
}

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

long fibonacci(long);

int main(int argc, char *argv[])
{
    int n;
    printf("Calcolo numero di Fibonacci - Digita un numero naturale: ");
    scanf("%d", &n);
    printf("Il numero di Fibonacci nella posizione %d e': %ld\n", n, fibonacci(n));
    system("PAUSE");
    return 0;
}

long fibonacci(long i)
{
    if (i == 0)
        return 0;
    else if (i == 1)
        return 1;
    else
        return fibonacci(i-1) + fibonacci(i-2);
}

```

Capitolo 13 - I puntatori

```

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

main()
{
    int vettore[10]={};
    int *indice;
    indice = &vettore[0];
    /*
    Le seguenti due righe di codice danno lo stesso risultato.
    Per verificarlo potete commentare alternativamente una delle due righe,
    compilare e verificare il risultato ottenuto a video.
    */
    vettore[1]=100;
    *(indice + 1)=100;

    printf("Test puntatore: %d\n", vettore[1]);

    system("pause");
    return 0;
}

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

main()
{
    int vettore[10]={};
    int *indice;
    indice = &vettore[0];

    printf("Test puntatore: %d\n", *indice + 1 );

    system("pause");
    return 0;
}

```

```

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

main()
{
    int i=0;

    char atest[]= "Ciccio Pasticcio";
    char *ptest = "ciccio pasticcio";

    printf("Stringa con array\n");
    while (atest[i] != '\0')
    {
        printf("%c", atest[i]);
        i++;
    }
    printf("\n\n");

    printf("Stringa con puntatore\n");
    while (*ptest != '\0')
    {
        printf("%c", *ptest);
        ptest++;
    }
    printf("\n\n");

    system("PAUSE");
    return 0;
}

```

```

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

```

```
int cambia(int);
```

```

int main(int argc, char *argv[])
{
    int a=0, b=0;
    printf("Il valore di a prima della chiamata e': %d\n", a );
    printf("Il valore di b prima della chiamata e': %d\n", b );
    b = cambia(a);
    printf("Il valore di a dopo la chiamata e': %d\n", a );
    printf("Il valore di b dopo la chiamata e': %d\n", b );

    system("PAUSE");
    return 0;
}

```

```

int cambia (int a)
{
    a = a+1;
    return a;
}

```

```

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

```

```
void cambia(int *);
```

```

int main(int argc, char *argv[])
{
    int a, b, *punt;
    punt=&a;
    a=0;
    printf("Il valore di a prima della chiamata e': %d\n", a );
    cambia(punt);
    printf("Il valore di a dopo la chiamata e': %d\n", a );

    system("PAUSE");
}

```

```

    return 0;
}

void cambia (int *x)
{
    *x= *x+1;
    return;
}

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

int strTest(char[], char[]);

int main(int argc, char *argv[])
{
    char a[10]="Pipp0";
    char b[10]="pIppo";
    int ret=0;
    ret = strTest(a, b);
    printf("Differenza tra stringhe: %d\n", ret);

    system("PAUSE");
    return 0;
}

int strTest(char x[], char y[])
{
    int i=0, cont=0;
    while ((a[i] != '\0') && (b[i] != '\0'))
    {
        if (a[i]!=b[i])
        {
            cont++;
        }
        i++;
    }
    return cont;
}

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

int main(int argc, char *argv[])
{
    int i;
    printf("argc e': %d\n", argc);
    for(i=0; i < argc; i++)
    {
        printf("Argomento: %s\n", argv[i]);
    }

    /* oppure */
    for(i=0; i < argc; i++)
    {
        printf("Argomento: %s\n", *argv);
        argv++;
    }

    system("PAUSE");
    return 0;
}

```

Capitolo 14 - Le strutture: molti dati ed un'unica variabile

```

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

int main(int argc, char *argv[])
{
    struct persona
    {
        char nome[20];
        char cognome[20];
        char indirizzo[30];
        int eta;
    };

    struct persona cliente;

    printf("Nome: ");
    gets(cliente.nome);
    printf("Cogome: ");
    gets(cliente.cognome);
    printf("Eta': ");
    scanf("%d", &cliente.eta);

    printf("\nIl nome e': %s", cliente.nome);
    printf("\nIl cognome e': %s", cliente.cognome);
    printf("\nL'eta' e': %d\n", cliente.eta);
    printf("\n");

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    struct data
    {
        int giorno;
        int mese;
        int anno;
    };

    struct persona
    {
        char nome[20];
        char cognome[20];
        char indirizzo[30];
        struct data nascita;
    };

    struct persona Topolino = {"Mickey", "Mouse", "Topolinia", 18,11,1928};

    printf("La data di nascita di %s %s e': %d/%d/%d ", Topolino.nome, Topolino.cognome,
    Topolino.nascita.giorno, Topolino.nascita.mese, Topolino.nascita.anno );

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

```

```

#include <stdio.h>
#include <stdlib.h>
#define MAX_CLIENTI 10

int main(int argc, char *argv[])
{
    struct data
    {
        int giorno;

```

```

    int mese;
    int anno;
};

struct persona
{
    char nome[20];
    char cognome[20];
    char indirizzo[30];
    struct data nascita;
};

struct persona cliente[MAX_CLIENTI];
int i;

//Inserimento dati
printf("Inserisci i dati dei tuoi clienti\n\n");
for (i=0; i<MAX_CLIENTI; i++)
{
    printf("Cliente: %d\n", i+1);
    printf("Nome: ");
    gets(cliente[i].nome);
    printf("Cognome: ");
    gets(cliente[i].cognome);
}

//Stampa dati
printf("\nL'elenco dei tuoi clienti e'\n\n");
for (i=0; i<MAX_CLIENTI; i++)
{
    printf("\nNome: %s", cliente[i].nome);
    printf("\nCognome: %s", cliente[i].cognome);
    printf("\n");
}

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

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

struct punto
{
    int x;
    int y;
    int z;
};

void stampa(struct punto);

int main(int argc, char *argv[])
{
    struct punto A;

    A.x=1;
    A.y=1;
    A.z=1;

    stampa(A);

    system("PAUSE");
    return 0;
}

void stampa(struct punto A)
{
    printf("Le coordinate del punto sono: %d - %d - %d\n", A.x, A.y, A.z);
}

```



```

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

struct punto
{
    int x;
    int y;
    int z;
};

void valorizza(struct punto *);
void stampa(struct punto *);

int main(int argc, char *argv[])
{
    struct punto A;

    valorizza(&A);
    stampa(&A);

    system("PAUSE");
    return 0;
}

void valorizza(struct punto *A)
{
    (*A).x = 2;
    (*A).y = 2;
    (*A).z = 2;
}

void stampa(struct punto *A)
{
    printf("Le coordinate del punto sono: %d - %d - %d\n", (*A).x, (*A).y, (*A).z);
}

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

typedef struct _punto
{
    int x;
    int y;
    int z;
} punto;

int main(int argc, char *argv[])
{
    punto A;

    A.x = 1;
    A.y = 1;
    A.z = 1;
    printf("Le coordinate del punto sono: %d - %d - %d\n", A.x, A.y, A.z);

    system("PAUSE");
    return 0;
}

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

void train_status(void);

struct train
{
    unsigned int moving : 1;
}

```

```

    unsigned int forward : 1;
};
struct train t;

int main(int argc, char *argv[])
{
    t.moving = 1;
    t.forward = 1;

    train_status();

    system("PAUSE");
    return 0;
}

```

```

void train_status(void)
{
    if (t.moving == 1)
    {
        printf("Il treno va...");
        if (t.forward == 1)
        {
            printf("in avanti!");
        }
        else
        {
            printf("indietro!");
        }
    }
    else
    {
        printf("Sono fermo!");
    }
}

```

```

#include <stdio.h>
#include <stdlib.h>
struct persona
{
    char nome[20];
    char cognome[25];
    char indirizzo[30];
};

```

```

int main(int argc, char *argv[])
{
    struct persona cliente;

    printf("Lo spazio di memoria in byte occupato dalla struttura e': %d\n", sizeof(cliente));

    system("PAUSE");
    return 0;
}

```

```

#include <stdio.h>
#include <stdlib.h>
union persona
{
    char nome[20];
    char cognome[25];
    char indirizzo[30];
};

```

```

int main(int argc, char *argv[])
{
    union persona cliente;

    printf("Lo spazio di memoria in byte occupato dalla unione e': %d\n", sizeof(cliente));
}

```

```

    system("PAUSE");
    return 0;
}

#include <stdio.h>
#include <stdlib.h>
union persona
{
    char nome[20];
    char cognome[25];
    char indirizzo[30];
};

int main(int argc, char *argv[])
{
    union persona cliente;

    strcpy(cliente.nome, "Carlo");
    strcpy(cliente.cognome, "Mazzone");
    strcpy(cliente.indirizzo, "Via Dei Viali");

    printf("Nome : %s\n", cliente.nome);
    printf("Cognome : %s\n", cliente.cognome);
    printf("Indirizzo : %s\n", cliente.indirizzo);

    system("PAUSE");
    return 0;
}

```

Capitolo 15 - I file e gli archivi di dati secondo il C

```

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

int main(int argc, char *argv[])
{
    FILE * fp;
    fp = fopen("carlo.txt", "w");
    fclose(fp);

    system("PAUSE");
    return 0;
}

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

int main(int argc, char *argv[])
{
    FILE * fp;
    if (argc != 2)
    {
        printf("Devi indicare il nome del file da creare!\n");
    }
    else
    {
        fp = fopen(argv[1], "w");
        fclose(fp);
        printf("Il file è stato creato!\n");
    }

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    FILE * fp;
    fp = fopen("carlo.txt", "w");
    fprintf(fp, "Carlo Mazzone");
    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    FILE * fp;
    fp = fopen("miofile.txt", "w");
    fprintf(fp, "Io sono la prima riga\n");
    fprintf(fp, "Io sono la seconda riga\n");
    fprintf(fp, "Io sono la terza ed ultima riga");
    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    FILE * fp;
    int i;
    fp = fopen("numeri.txt", "w");
    for (i=1; i<=20; i++)
    {
        fprintf(fp, "%d\n", i);
    }

    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    FILE * fp;
    int x;
    fp = fopen("numeri.txt", "r");
    while (!feof(fp))
    {
        fscanf(fp, "%d", &x);
        printf("%d ", x);
    }

    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    FILE * fp;
    int x;
    if ( (fp = fopen("numeri.txt", "r")) == NULL )
    {
        printf("Si e' verificato un errore nell'apertura del file.");
        return 1;
    }
    while (!feof(fp))
    {
        fscanf(fp, "%d", &x);
        printf("%d ", x);
    }

    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    FILE * fp;
    char c;
    if ( (fp = fopen("miofile.txt", "r")) == NULL )
    {
        printf("Si è verificato un errore nell'apertura del file.");
        return 1;
    }
    while ((c = fgetc(fp))!= EOF )
    {
        printf("%c ", c);
    }

    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

```

```

int main(int argc, char *argv[])
{
    FILE * fp;
    char c;

    if ( (fp = fopen("miofile.txt", "w")) == NULL )
    {
        printf("Si è verificato un errore nell'apertura del file.");
        return 1;
    }

    while ((c = getchar()) != '\n')
    {
        fputc(c, fp);
    }

    fclose(fp);

    system("PAUSE");
    return 0;
}

```

```

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

int main(int argc, char *argv[])
{
    FILE * fp;
    char x[]="Carlo Mazzone";

    if ( (fp = fopen("miofile.bin", "wb")) == NULL )
    {
        printf("Si è verificato un errore nell'apertura del file.");
        return 1;
    }

    fwrite(x, sizeof(x[0]), 13, fp);

    fclose(fp);

    system("PAUSE");
    return 0;
}

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

int main(int argc, char *argv[])
{
    FILE * fp;
    int x[]={1,2,3,4,5};
    int y[5];

    if ( (fp = fopen("miofile.bin", "wb+")) == NULL )
    {
        printf("Si è verificato un errore nell'apertura del file.");
        return 1;
    }

    fwrite(x, sizeof(x[0]), 5, fp);
    printf("La dimensione di una cella e': %d \n",sizeof(x[0]));
    printf("Il cursore dopo la scrittura e' in posizione: %d \n", ftell(fp));

    fseek(fp,0, SEEK_SET);

    printf("Il cursore e' ora in posizione: %d \n",ftell(fp));

    fread(y, sizeof(int), 5 ,fp);

    fclose(fp);

    printf("Il contenuto del file e': ");

    int i;
    for (i=0; i <5; i++)
    {
        printf("%d ", y[i]);
    }
    printf("\n");

    system("PAUSE");
    return 0;
}

```

Capitolo 16 - La programmazione orientata agli oggetti e lo sviluppo di software complesso

...

Capitolo 17 - Dal C al C++

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
{
    cout << "Hello world!" << endl;
    return 0;
}
```

```
#include <iostream>
```

```
int main()
{
    std::cout << "Salve mondo!" << std::endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
namespace Pippo
{
    int x, y;
}
```

```
int main()
{
    Pippo::x = 1;
    Pippo::y = 2;

    cout << "I valori di x e y sono: " << Pippo::x << " " << Pippo::y << endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
namespace Pippo
{
    int x = 1;
    int y = 2;
}
```

```
namespace Pluto
{
    int x = 3;
    int y = 4;
}
```

```
int main()
{
    cout << "Prima variabile x: " << Pippo::x << endl;
    cout << "Seconda variabile x: " << Pluto::x << endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
namespace Pippo
```

```

{
    int x = 1;
    int y = 2;
}

namespace Pluto
{
    int x = 3;
    int y = 4;
}

using Pippo::x;
int main()
{
    cout << "Prima variabile x: " << x << endl;
    cout << "Seconda variabile x: " << Pluto::x << endl;

    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```

namespace Pippo
{
    int x = 1;
    int y = 2;
}

```

```

namespace Pluto
{
    int x = 3;
    int y = 4;
}

```

```

int main()
{
    using namespace Pippo;

    cout << "Prima variabile x: " << x << endl;
    cout << "Prima variabile y: " << y << endl;
    cout << "Seconda variabile x: " << Pluto::x << endl;

    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```

main()
{
    cout << "Salve mondo\n";
    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```

namespace Pippo
{
    int x = 1;
    int y = 2;
}

```

```

namespace Pluto
{
    int x = 3;
    int y = 4;
}

```



```

}

int main()
{
    {
        using namespace Pippo;
        cout << "Prima variabile x: " << x << endl;
    }
    {
        using namespace Pluto;
        cout << "Seconda variabile x: " << x << endl;
    }

    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```
int x = 1;
```

```

int main()
{
    int x = 2;
    {
        int x = 3;
        cout << "x: " << x << endl;
    }

    cout << "x: " << x << endl;
    cout << "x: " << ::x << endl;

    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```

int main()
{
    cout << "Riferimenti!" << endl;
    int x = 1;
    int &r = x;
    r++;
    cout << "r: " << r << endl;

    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```
void scambia(int , int);
```

```

int main()
{
    cout << "Scambiamoci i valori" << endl;

    int a = 1;
    int b = 2;
    scambia(a, b);

    cout << "a: " << a << endl;
    cout << "b: " << b << endl;

    return 0;
}

```

```

void scambia (int x, int y)
{
    int z;

    z=x;
    x=y;
    y=z;
}

#include <iostream>
using namespace std;

void scambia(int& , int&);

int main()
{
    cout << "Scambiamoci i valori" << endl;

    int a = 1;
    int b = 2;
    scambia(a, b);

    cout << "a: " << a << endl;
    cout << "b: " << b << endl;

    return 0;
}

void scambia (int& x, int& y)
{
    int z;

    z = x;
    x = y;
    y = z;
}

```

Capitolo 18 - Input ed output di dati

```

#include <iostream>
using namespace std;

main()
{
    int x, s;
    cout << "Dammi un numero: ";
    cin >> x;
    s = x + 1;
    cout << "Il successivo e': " << s;

    return 0;
}

```

```

#include <iostream>
using namespace std;

main()
{
    int x, y, s;
    cout << "Dammi un numero: ";
    cin >> x;
    cout << "Dammi un altro numero: ";
    cin >> y;
}

```

```
s = x + y;
cout << "La somma e': " << s;

return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
main()
{
    int x, y, s;
    cout << "Inserisci due numeri: ";
    cin >> x >> y;
    s = x + y;
    cout << "La somma e': " << s;

    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
int main()
{
    char s[25];
    cout << "Dimmi come ti chiami (e ti diro' chi sei): ";
    cin >> s;
    cout << "Tu sei " << s << endl;

    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
int main()
{
    char s1[25],s2[25];
    cout << "Stringa 1: ";
    cin >> s1;
    cout << s1 << endl;
    cout << "Stringa 2: ";
    cin >> s2;
    cout << s2 << endl;

    return 0;
}
```

```
#include <iostream>
using namespace std;
```

```
int main()
{
    char s1[25],s2[25];
    cout << "Stringa 1: ";
    cin.getline(s1, 25);
    cout << s1 << endl;
    cout << "Stringa 2: ";
    cin.getline(s2, 25);
    cout << s2 << endl;

    return 0;
}
```

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
{
    char numero, s[25];
    cout << "Dammi il numero: ";
    cin >> numero;
    cout << "Dammi la stringa: ";
    cin.getline(s, 25);
    cout << numero << endl;
    cout << s << endl;

    return 0;
}
```

```
#include <iostream>
#include <iomanip>
using namespace std;
```

```
int main()
{
    cout << "12345678901234567890" << endl;

    cout << setw(15) << "Carlo" << endl;
    cout << setw(15) << "Pippo" << endl;
    cout << setw(15) << "Pluto" << endl;
    cout << setw(15) << "Topolino" << endl;

    return 0;
}
```

```
#include <iostream>
#include <iomanip>
using namespace std;
```

```
int main()
{
    cout << setfill('*') << endl;
    cout << setw(80) << " " << endl;

    return 0;
}
```

```
#include <iostream>
#include <iomanip>
using namespace std;
```

```
int main()
{
    const double x = 3.14159;
    cout << setprecision(3) << x << endl;

    return 0;
}
```

```
#include <iostream>
#include <iomanip>
using namespace std;
```

```
int main()
{
    cout << fixed;
    cout << setprecision(2) << 123.456 << endl;
    cout << setprecision(3) << 123.456 << endl;
    cout << setprecision(4) << 123.456 << endl;

    return 0;
}
```

Capitolo 19 - Un oggetto davvero di “classe”

```
#include <iostream>
using namespace std;

struct Rettangolo
{
    int x;
    int y;
};

int main()
{
    //inizializzo la struct Rettangolo
    Rettangolo rettangolo1;

    //assegno valori a base ed altezza
    rettangolo1.x = 20;
    rettangolo1.y = 10;

    //stampo valori
    cout << "Base: " << rettangolo1.x << endl;
    cout << "Altezza: " << rettangolo1.y << endl;

    return 0;
}

#include <iostream>
using namespace std;

class Rettangolo
{
public:
    int x;
    int y;
    int Area();
};

int Rettangolo::Area()
{
    return x * y;
}

int main()
{
    cout << "Rettangoli a confronto" << endl;
    Rettangolo rettangolo1;
    Rettangolo rettangolo2;

    //imposto valori
    rettangolo1.x = 10;
    rettangolo1.y = 6;
    rettangolo2.x = 8;
    rettangolo2.y = 4;
    cout << "L'area del primo rettangolo vale: " << rettangolo1.Area() << endl;
    cout << "L'area del secondo rettangolo vale: " << rettangolo2.Area() << endl;

    return 0;
}
```

Capitolo 20 - L'arte del creare e del distruggere

```
#include <iostream>
using namespace std;

class Rettangolo
{
public:
    int x;
    int y;
    Rettangolo(int, int);
    int Area();
};

Rettangolo::Rettangolo(int a, int b)
{
    x = a;
    y = b;
}

int Rettangolo::Area()
{
    return x * y;
}

int main()
{
    cout << "La mia classe rettangolo" << endl;
    Rettangolo rettangolo1 (3, 6);
    Rettangolo rettangolo2 (3, 17);

    cout << "L'area del primo rettangolo vale: " << rettangolo1.Area() << endl;
    cout << "L'area del secondo rettangolo vale: " << rettangolo2.Area() << endl;

    return 0;
}

//COMPILAZIONE SEPARATA
//File main.cpp
#include <iostream>
#include "Rettangolo.h"

using namespace std;

int main()
{
    cout << "Rettangoli e moduli" << endl;

    Rettangolo rettangolo1 (2, 4);
    cout << "L'area del rettangolo e': " << rettangolo1.Area() << endl;

    return 0;
}

//File Rettangolo.h
#ifndef RETTANGOLO_H_INCLUDED
#define RETTANGOLO_H_INCLUDED

class Rettangolo
{
public:
    int x;
    int y;
    Rettangolo(int, int);
    ~Rettangolo();
    int Area();
};
```

```

#endif // RETTANGOLO_H_INCLUDED

//File Rettangolo.cpp
#include <iostream>
#include "Rettangolo.h"

using namespace std;

Rettangolo::Rettangolo(int a, int b)
{
    x = a;
    y = b;
}

Rettangolo::~Rettangolo()
{
    cout << "Fine operazioni";
}

int Rettangolo::Area()
{
    return x * y;
}

```

Capitolo 21 - Polimorfismo, ovvero come avere differenti comportamenti e rimanere credibili

```

#include <stdio.h>
#include <stdlib.h>
void Tab(int);

int main()
{
    Tab(3);
    printf("Test tabulazioni C!\n");

    return 0;
}

void Tab (int n)
{
    int i;

    for (i=1; i<=n; i++)
    {
        printf("\t");
    }
}

#include <iostream>

using namespace std;

void Tab(void);
void Tab(int);

int main()
{
    Tab();
    cout << "Test" << endl;
    Tab(2);
    cout << "tabulazioni" << endl;

    return 0;
}

```

```

}

void Tab (void)
{
    cout << "\t";
}

void Tab (int n)
{
    int i;

    for (i=1; i<=n; i++)
    {
        cout << "\t";
    }
}

#include <iostream>
using namespace std;

class Rettangolo
{
public:
    int x;
    int y;
    Rettangolo(int, int);
    Rettangolo();
    int Area();
};

Rettangolo::Rettangolo(int a, int b)
{
    x=a;
    y=b;
}

Rettangolo::Rettangolo()
{
    x=2;
    y=1;
}

int Rettangolo::Area()
{
    return x * y;
}

int main()
{
    cout << "La mia classe rettangolo" << endl;
    Rettangolo rettangolo1 (3, 6);
    Rettangolo rettangolo2;

    cout << "L'area del primo rettangolo vale: " << rettangolo1.Area() << endl;
    cout << "L'area del secondo rettangolo vale: " << rettangolo2.Area() << endl;

    return 0;
}

#include <iostream>
using namespace std;

class Calcolatrice
{
public:
    int x;
    int y;
}

```



```

    void Dividi(int, int);
    void Dividi(double, double);
};

void Calcolatrice::Dividi(int a, int b)
{
    cout << a/b;
}

void Calcolatrice::Dividi(double a, double b)
{
    cout << a/b;
}

```

```

#include <iostream>
using namespace std;

```

```

class Point
{
    public:

    float x;
    float y;
    Point *next;

    Point();
    Point(float, float);
};

```

```

Point::Point()
{
    x=0;
    y=0;
}

```

```

Point::Point(float a, float b)
{
    x=a;
    y=b;
}

```

```

int main()
{
    cout << "Segmenti!" << endl;
    return 0;
}

```

```

#include <iostream>
using namespace std;

```

```

class Point
{
    public:

    float x;
    float y;
    Point *next;

    Point();
    Point(float, float);
    Point operator+ (Point);
};

```

```

Point::Point()
{
    x=0;
    y=0;
}

```

```

Point::Point(float a, float b)
{
    x = a;
    y = b;
}

Point Point::operator+ (Point p)
{
    Point s;

    s.x = x + p.x;
    s.y = y + p.y;

    return s;
}

int main()
{
    cout << "Somma di punti" << endl;

    Point a(1,1);
    Point b(3,1);

    Point s;

    s = a + b;

    cout << "Le coordinate dei punti sono: " << endl;
    cout << "Punto a: " << a.x << " " << a.y << endl;
    cout << "Punto b: " << b.x << " " << b.y << endl;
    cout << "Punto s: " << s.x << " " << s.y << endl;

    return 0;
}

```

Capitolo 22 – Ereditarietà

```

#include <iostream>
using namespace std;

class Cerchio
{
public:
    float raggio;
    float Area();
    Cerchio (float);
};

Cerchio::Cerchio(float r)
{
    raggio = r;
}

float Cerchio::Area()
{
    return raggio * raggio * 3.14;
}

int main()
{
    cout << "Test ereditarieta'" << endl;
    Cerchio cerchiol(4.0);

    cout << "L'area del cerchio e': " << cerchiol.Area() << endl;
}

```

```

    return 0;
}

#include <iostream>
using namespace std;

class Cerchio
{
    public:
    float raggio;
    float Area();
    Cerchio ();
    Cerchio (float);
};

Cerchio::Cerchio()
{
    raggio=1;
}

Cerchio::Cerchio(float r)
{
    raggio=r;
}

float Cerchio::Area()
{
    return raggio * raggio * 3.14;
}

class Cilindro: public Cerchio
{
    public:
    float altezza;
    float Volume();
    Cilindro ();
    Cilindro (float, float);
};

Cilindro::Cilindro()
{
    altezza= 1;
}

Cilindro::Cilindro(float r, float h)
{
    raggio=r;
    altezza= h;
}

float Cilindro::Volume()
{
    return raggio * raggio * 3.14 * altezza;
}

int main()
{
    Cilindro cilindro1;
    Cilindro cilindro2(2,2);

    cout << "I DUE CILINDRI" << endl;

    cout << "L'area della base del primo cilindro e': " << cilindro1.Area() << endl;
    cout << "Il volume del primo cilindro e': " << cilindro1.Volume() << endl;
}

```

```

    cout << "L'area della base del secondo cilindro e': " << cilindro2.Area() << endl;
    cout << "Il volume del secondo cilindro e': " << cilindro2.Volume() << endl;

    return 0;
}

#include <iostream>
using namespace std;

//classe Punto
class Punto
{
    public:
    void Disegna();
};

void Punto::Disegna()
{
    cout << "." << endl;
}

//Classe Linea
class Linea: public Punto
{
    public:
    void Disegna();
};

void Linea::Disegna()
{
    cout << "_____" << endl;
}

//Classe Tratteggio
class Tratteggio: public Punto
{
    public:
    void Disegna();
};

void Tratteggio::Disegna()
{
    cout << "-----" << endl;
}
}

```

Capitolo 23 – La programmazione generica e la libreria standard del C++

```

#include <iostream>
using namespace std;

template <class T>
class Rettangolo
{
    public:
    T x;
    T y;
    Rettangolo(T, T);
    T Area();
};

```

```

template <class T>
Rettangolo<T>::Rettangolo (T a, T b)
{
    x = a;
    y = b;
}

template <class T>
T Rettangolo<T>::Area ()
{
    T ris;
    ris = x * y;
    return ris;
}

```

Capitolo 24 – Flussi e file nel C++

```

#include <iostream>
#include <fstream>

using namespace std;

int main()
{
    ofstream f;
    f.open ("primo.txt");
    f.close();

    cout << "Fine operazioni!" << endl;
    return 0;
}

#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    ofstream f;
    f.open ("anagrafica.txt");
    f << "Nome\tCognome\temail\n";
    f << "Carlo\tMazzone\tinfo@tesseract.it\n";
    f << "Ciccio\tPasticcio\tciccio@example.com\n";
    f << "Mickey\tMouse\tmickey@example.com\n";
    f.close();

    cout << "Fine registrazione!" << endl;
    return 0;
}

#include <iostream>
#include <fstream>
using namespace std;

int main()
{
    ifstream f;
    f.open("anagrafica.txt", ios::in);

    if(!f)
    {
        cout << "Il file specificato non esiste!";
        return 1;
    }

    string linea;

```

```

while(!f.eof())
{
    getline(f, linea);
    cout << linea << endl;
}
f.close();

return 0;

}

#include <iostream>
using namespace std;

void Inserisci();
void Lista();
void Modifica();

int main()
{
    int x;
    cout << "I MIEI FUMETTI ver 0.1" << endl << endl;

    do
    {
        cout << "1 - Lista" << endl;
        cout << "2 - Inserisci" << endl;
        cout << "3 - Modifica" << endl;
        cout << "" << endl;
        cout << "0 - ESCI" << endl;
        cout << "" << endl;
        cout << "Operazione: ";
        cin >> x;

        switch(x)
        {
            case 1:
                Lista();
                break;

            case 2:
                Inserisci();
                break;

            case 3:
                Modifica();
                break;

            break;
        }
    }while (x != 0);

    return 0;
}

void Lista()
{
    cout << "LISTA FUMETTI" << endl;
    return;
}

void Inserisci()
{
    cout << "INSERIMENTO FUMETTO" << endl;
    return;
}

void Modifica()
{

```

```

    cout << "MODIFICA FUMETTO" << endl;
    return;
}

```

Capitolo 25 – La gestione degli errori e delle eccezioni

```

#include <iostream>

using namespace std;

int main()
{
    int a,b;
    cout << "Primo numero: ";
    cin >> a;
    cout << "Secondo numero: ";
    cin >> b;

    try
    {
        if (b==0)
        {
            throw 1;
        }
        else
        {
            cout << "Risultato della divisione: " << a/b << endl;
        }
    }

    catch (int e)
    {
        cout << "Eccezione catturata con numero di errore: " << e << endl;
    }

    return 0;
}

```

```

#include <iostream>
using namespace std;

int main()
{
    int a, b;
    cout << "Primo numero: ";
    cin >> a;
    cout << "Secondo numero: ";
    cin >> b;

    try
    {
        if (b==0)
        {
            throw "Divisione per zero!";
        }
        else
        {
            cout << "Risultato della divisione: " << a/b << endl;
        }
    }

    catch (const char* msg)
    {
        cout << "Eccezione catturata con messaggio: " << msg << endl;
    }
}

```

```

    }

    return 0;
}

#include <iostream>
using namespace std;

int main()
{
    int a, b;
    cout << "Primo numero: ";
    cin >> a;
    cout << "Secondo numero: ";
    cin >> b;

    try
    {
        if (b==0)
        {
            throw 1.0;
        }
        else
        {
            cout << "Risultato della divisione: " << a/b << endl;
        }
    }
    catch (int e)
    {
        cout << "Eccezione catturata con numero di errore: " << e << endl;
    }
    catch (const char* msg)
    {
        cout << "Eccezione catturata con messaggio: " << msg << endl;
    }
    catch (...)
    {
        cout << "Eccezione generica."<< endl;
    }

    return 0;
}

```

Capitolo 26 – L’allocazione dinamica della memoria

```

#include <iostream>
using namespace std;

int main()
{
    cout << "L'impossibilità di gestire le necessità del momento!" << endl;
    int n;
    cout << "Di quanto spazio hai bisogno?";
    cin >> n;
    int a[n]; //non consentito

    return 0;
}

```



```

#include <iostream>
using namespace std;

int main()
{
    cout << "Una singola variabile dinamica" << endl;
    int *p = new int;
    *p = 1;
    cout << "Il valore e': " << *p << endl;
    delete p;

    return 0;
}

```

```

#include <iostream>
using namespace std;

int main()
{
    cout << "Allocazione dinamica" << endl;
    int n;
    cout << "Di quanto spazio hai bisogno?";
    cin >> n;

    //allocazione array
    int *p = new int[n];

    //valorizzazione array
    int i;
    for (i=0; i <n; i++)
    {
        p[i]=i+1; //l'array parte da zero ma noi inseriamo i numeri a partire da 1 (quindi i+1)
    }

    //stampa array
    for (i=0; i <n; i++)
    {
        cout << p[i] << " ";
    }

    //deallocazione
    delete[] p;

    return 0;
}

```

```

#include <iostream>
using namespace std;

int main()
{
    int *p;
    p = new (nothrow) int [1000000000];
    if (p == 0)
    {
        cout << "Allocazione FALLITA!";
    }
    else
    {
        cout << "Allocazione riuscita";
    }

    return 0;
}

```

```

#include <iostream>

```

```

#include <exception>
using namespace std;

int main()
{
    try
    {
        // codice
        int *p;
        p = new int [1000000000];
    }
    catch (exception& e)
    {
        // Gestione eccezione e
        cout << "Allocazione FALLITA! " ;
        cout << "Eccezione standard: " << e.what() << endl;
    }

    return 0;
}

```

```

#include <iostream>
using namespace std;

class Rettangolo
{
public:
    int x;
    int y;
    Rettangolo(int, int);
    int Area();
};

Rettangolo::Rettangolo(int a, int b)
{
    x = a;
    y = b;
}

int Rettangolo::Area()
{
    return x * y;
}

int main()
{
    cout << "La mia classe rettangolo e' dinamica" << endl;
    Rettangolo *pR1 = new Rettangolo (4, 7);

    cout << "L'area del rettangolo vale: " << pR1->Area() << endl;

    delete pR1;

    return 0;
}

```

Capitolo 27 – Le interfacce grafiche e la programmazione guidata da eventi

```

#include <windows.h>

/* Declare Windows procedure */
LRESULT CALLBACK WindowProcedure (HWND, UINT, WPARAM, LPARAM);

```

```

/* Make the class name into a global variable */
char szClassName[ ] = "CodeBlocksWindowsApp";

int WINAPI WinMain (HINSTANCE hThisInstance,
                   HINSTANCE hPrevInstance,
                   LPSTR lpszArgument,
                   int nCmdShow)
{
    HWND hwnd;           /* This is the handle for our window */
    MSG messages;       /* Here messages to the application are saved */
    WNDCLASSEX wincl;   /* Data structure for the windowclass */

    /* The Window structure */
    wincl.hInstance = hThisInstance;
    wincl.lpszClassName = szClassName;
    wincl.lpfnWndProc = WindowProcedure; /* This function is called by windows */
    wincl.style = CS_DBLCLKS;           /* Catch double-clicks */
    wincl.cbSize = sizeof (WNDCLASSEX);

    /* Use default icon and mouse-pointer */
    wincl.hIcon = LoadIcon (NULL, IDI_APPLICATION);
    wincl.hIconSm = LoadIcon (NULL, IDI_APPLICATION);
    wincl.hCursor = LoadCursor (NULL, IDC_ARROW);
    wincl.lpszMenuName = NULL;         /* No menu */
    wincl.cbClsExtra = 0;              /* No extra bytes after the window class */
    wincl.cbWndExtra = 0;              /* structure or the window instance */
    /* Use Windows's default colour as the background of the window */
    wincl.hbrBackground = (HBRUSH) COLOR_BACKGROUND;

    /* Register the window class, and if it fails quit the program */
    if (!RegisterClassEx (&wincl))
        return 0;

    /* The class is registered, let's create the program*/
    hwnd = CreateWindowEx (
        0, /* Extended possibilites for variation */
        szClassName, /* Classname */
        "Code::Blocks Template Windows App", /* Title Text */
        WS_OVERLAPPEDWINDOW, /* default window */
        CW_USEDEFAULT, /* Windows decides the position */
        CW_USEDEFAULT, /* where the window ends up on the screen */
        544, /* The programs width */
        375, /* and height in pixels */
        HWND_DESKTOP, /* The window is a child-window to desktop */
        NULL, /* No menu */
        hThisInstance, /* Program Instance handler */
        NULL /* No Window Creation data */
    );

    /* Make the window visible on the screen */
    ShowWindow (hwnd, nCmdShow);

    /* Run the message loop. It will run until GetMessage() returns 0 */
    while (GetMessage (&messages, NULL, 0, 0))
    {
        /* Translate virtual-key messages into character messages */
        TranslateMessage(&messages);
        /* Send message to WindowProcedure */
        DispatchMessage(&messages);
    }

    /* The program return-value is 0 - The value that PostQuitMessage() gave */
    return messages.wParam;
}

/* This function is called by the Windows function DispatchMessage() */
LRESULT CALLBACK WindowProcedure (HWND hwnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    switch (message) /* handle the messages */

```

```

{
    case WM_DESTROY:
        PostQuitMessage (0);          /* send a WM_QUIT to the message queue */
        break;
    default:
        /* for messages that we don't deal with */
        return DefWindowProc (hwnd, message, wParam, lParam);
}

return 0;
}

#define IDI_APP_ICONA    1

#define IDM_MAINMENU    100
#define IDM_FILE_NUOVO  101
#define IDM_FILE_APRI   102
#define IDM_FILE_SALVA  103
#define IDM_FILE_ESCI   104
#define IDM_AIUTO_INFO  105

#include "resource.h"

IDI_APP_ICONA    ICON    "app.ico"

IDM_MAINMENU MENU
BEGIN
    POPUP "File"
    BEGIN
        MENUITEM "Nuovo", IDM_FILE_NUOVO
        MENUITEM "Apri", IDM_FILE_APRI
        MENUITEM "Salva", IDM_FILE_SALVA
        MENUITEM SEPARATOR
        MENUITEM "Esci", IDM_FILE_ESCI
    END
    POPUP "Aiuto"
    BEGIN
        MENUITEM "Informazioni su...", IDM_AIUTO_INFO
    END
END
END

```