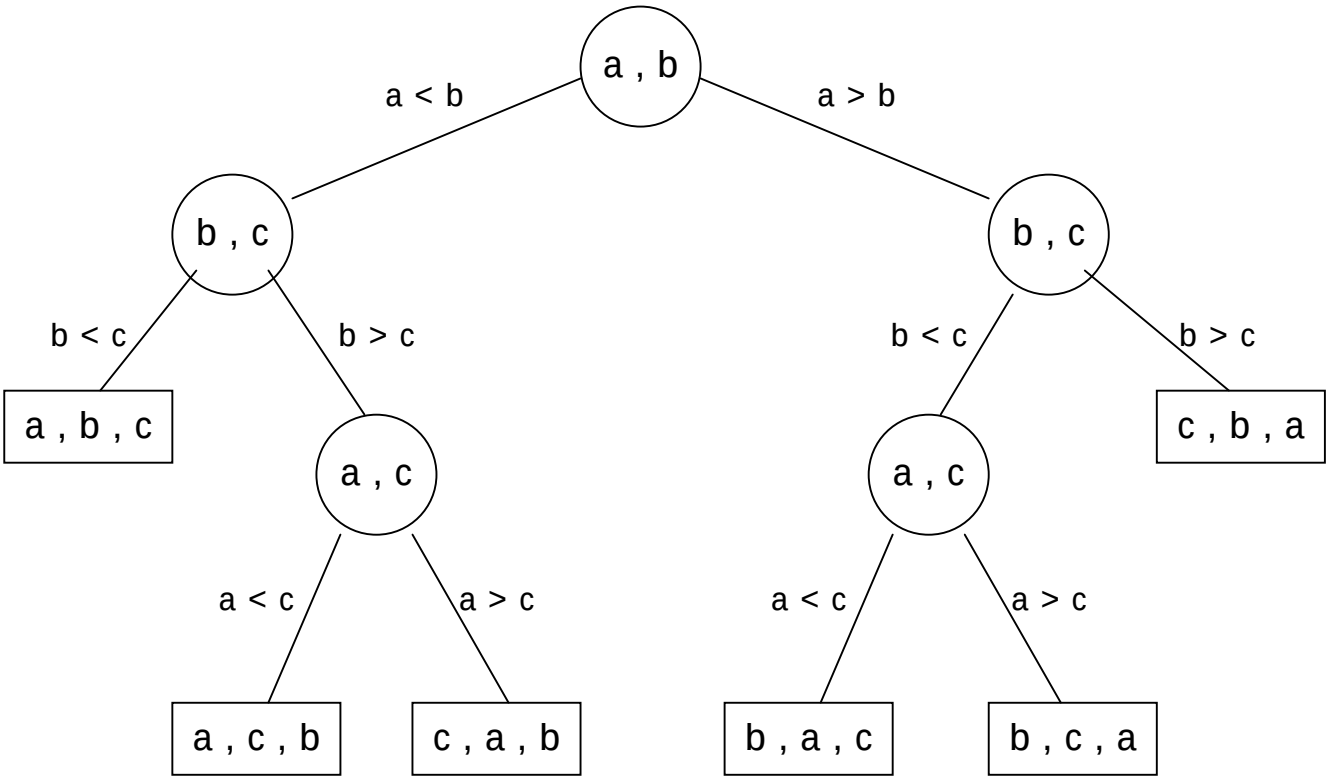
ESERCIZI DI PROGRAMMAZIONE C

ESERCIZIO 1 (Costrutti condizionali + Algoritmi)

Scrivere un programma che risolva il seguente problema.
Letti tre numeri interi *a, b, c* dallo standard input, stampare a terminale la sequenza dei tre numeri in ordine non decrescente.
Esempio: a = 10, b = 7, c = 9 deve dare in uscita 7 9 10.

1° soluzione

Stesura informale dell'algoritmo:
Si può sicuramente condurre un'analisi per casi e, in base a quello individuato, scrivere le tre variabili in ordine non decrescente.



```
#include <iostream.h> // inclusione della libreria standard
#include <stdlib.h>
int main( )
{
  int a,b,c;       // dichiarazione delle variabili

// legge tre interi a,b,c dallo standard input
  cout << "\n Inserisci il numero a: ";
  cin >> a;
  cout << "\n Inserisci il numero b: ";
  cin >> b;
  cout << "\n Inserisci il numero c: ";
  cin >> c;

  if (a < b) {

   if (b < c) {
      cout << "\n L'ordine voluto e': ";
      cout << a << " " << b << " " << c;
   }
   else {
      if (a < c) {
         cout << "\n L'ordine voluto e': ";
         cout << a << " " << c << " " << b;
      }
      else {
            cout << "\n L'ordine voluto e': ";
            cout << c << " " << a << " " << b;
      }
   }
  }
  else {

    if (c < b) {
      cout << "\n L'ordine voluto e': ";
      cout << c << " " << b << " " << a;
    }
    else {
      if (a < c) {
         cout << "\n L'ordine voluto e': ";
         cout << b << " " << a << " " << c;
      }
      else {
         cout << "\n L'ordine voluto e': ";
         cout << b << " " << c << " " << a;
      }
    }
  }
  cout << "\n\n FINE ELABORAZIONE....\n";
  system("PAUSE");
  return 0;
}
```

2°soluzione

Una seconda strategia di soluzione facilmente generalizzabile consiste nello scambiare ordinatamente le tre variabili finché i loro contenuti non risultino ordinati.
A tal fine quindi sarà necessaria sicuramente almeno un'altra variabile intera strumentale allo scambio tra due variabili.

Stesura informale dell'algoritmo:

1. Inizio del programma
2. Leggi i tre numeri *a, b, c*
3. confronta i valori di *a* e *b*, se non sono ordinati si effettua lo scambio
4. confronta i valori di *a* e *c*, se non sono ordinati si effettua lo scambio
5. confronta i valori di *b* e *c*, se non sono ordinati si effettua lo scambio
6. Stampa a video delle tre variabili *a, b, c*

Nota: i punti **3**, **4**, **5** risolvono ciascuno lo stesso tipo di *sottoproblema* che è quello di ordinare i valori di due variabili.
La seconda soluzione pur essendo più semplice della soluzione 1, dal punto di vista della struttura e quindi anche della leggibilità, è meno efficiente: effettua sempre tre confronti diversamente dalla prima soluzione che ne effettua due in due casi su sei.
A ben vedere quindi la soluzione 1 risulta essere la soluzione *ottima*: non è possibile trovarne una che effettui un numero inferiore di confronti.

```
#include <iostream.h> // inclusione della libreria standard
#include <stdlib.h>
int main( )
{
  int a,b,c,t; // dichiarazione delle variabili

// legge tre interi a,b,c dallo standard input

  cout << "\n Inserisci il numero a: ";
  cin >> a;
  cout << "\n Inserisci il numero b: ";
  cin >> b;
  cout << "\n Inserisci il numero c: ";
  cin >> c;

  // ordinamento dei valori delle variabili a,b
  if (a > b)
  {
    // Scambio dei valori delle due variabili a,b
    t = a;
    a = b;
    b = t;
  }
  // ordinamento dei valori delle variabili a,c
  if (a > c)
  {
    // Scambio dei valori delle due variabili a,c
    t = a;
    a = c;
    c = t;
  }
  // la variabile a contiene ora sicuramente il
  // valore più piccolo tra quelli inseriti.

  // ordinamento dei valori delle variabili b,c
  if (b > c)
  {
    // Scambio dei valori delle due variabili b,c
    t = b;
    b = c;
    c = t;
  }
  cout << "\n L'ordine voluto e': ";
  cout << a << " " << b << " " << c;

  cout << "\n\nFINE ELABORAZIONE\n";
  system("PAUSE");
  return 0;
} // end main
```

ESERCIZIO 2 (Costrutti condizionali + Casting delle variabili)

Realizzare un programma che, dato in ingresso un angolo specificato in gradi come un numero intero, fornisca la relativa conversione in radianti. L'angolo deve essere compreso tra 0 e 360 gradi, altrimenti il programma stampa un messaggio di errore e termina.

Nota: Si consideri 3.14 il valore di pi greco.

```
#include <iostream.h>
#include <stdlib.h>
#define PIGRECO 3.14 // uso la direttiva #define per la costante PIGRECO

int main()
{
  int gradi;
  float rad;
  cout << "\nInserire un angolo compreso tra 0 e 360 gradi: ";
  cin >> gradi;

  if (gradi < 0 || gradi > 360) // numero fuori range
  {
    cout << "\nErrore: e' stato inserito un angolo fuori range [0,360]";
  }
  else
  {
    rad = ((float) gradi) * PIGRECO / 180.0; // uso la costante PIGRECO
    cout << "\n\nL'angolo inserito di " << gradi;
    cout << " gradi corrisponde a " << rad <<" radianti";
  }

  cout << "\n\nFINE ELABORAZIONE -------------------\n";


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

Esercizio 3

Scrivere un programma che proponga all'utente di inserire un intervallo di temperature espresse in gradi Celsius. Se dopo l'inserimento di entrambi gli estremi una temperatura è inferiore a -273,15°C, il programma termina. Successivamente, stampare a terminale la conversione dell'intervallo in gradi Fahrenheit.

Nota: Fahrenheit = 32+9/5 * Celsius.

```
#include <stdlib.h>
#include <iostream.h>
int main( )
{

  float inf, sup;

  cout << "************************************************" << endl;
  cout << "******** Inserire il limite inferiore, inf:  ";
  cin >> inf;
  cout << "************************************************" << endl << endl;
  cout << "************************************************" << endl;
  cout << "******** Inserire il limite superiore, cmax=:  ";
  cin >> sup;
  cout << "************************************************" << endl << endl;
  if ((inf >=-273.15)&&(inf <=sup)) {
    cout << "[ "<<inf <<", " <<sup<<" ]°C";
    cout << "---";
    cout << "[ " << (9/5.0 * inf + 32) << ", " << (9.0/5.0 * sup + 32) << "]°F";
  }
  else cout << “Errore nell’immissione dei dati!” << endl;

  cout << "\n\nFINE ELABORAZIONE...\n";

  system("PAUSE");
  return 0;
} // end main
```

ESERCIZIO 4 (Creazione di un menù + Costrutti condizionali)

Realizzare un programma che legga tre valori interi (n1, n2, n3) compresi tra 1 e 100, estremi inclusi, e poi presenti a video il seguente menù di operazioni possibili:
A - somma tra n1, n2 e n3
B - prodotto tra n1 e n2
C - sottrazione tra n3 e n1
D - divisione tra n1 e n2 (risultato double).
X - uscita dal programma

Legge poi un carattere da tastiera: se il carattere è tra quelli indicati nel menù, si deve eseguire l'operazione richiesta, stampare i numeri utilizzati nell'operazione e il risultato e poi ripresentare il menù altrimenti il carattere deve essere ignorato e si deve ripresentare solo il menù. Nel caso il carattere sia X, il programma termina.

Nota: nel caso D è necessaria la conversione di almeno uno dei due operandi.

```
#include <iostream.h>
#include <stdlib.h>
int main()
{
  int n1, n2, n3, intres;
  double dblres;
  char c;

  do  // inserimento primo numero
  {
    cout << "\nInserire il PRIMO numero intero >= 1 e <= 100: ";
    cin >> n1;
  } while (n1 < 1 || n1 > 100);
  cout << "\n-------------------------\n";

  do  // inserimento secondo numero
  {
    cout << "\nInserire il SECONDO numero intero >= 1 e <= 100: ";
    cin >> n2;
  } while (n2 < 1 || n2 > 100);
  cout << "\n-------------------------\n";

  do  // inserimento terzo numero
  {
    cout << "\nInserire il TERZO numero intero >= 1 e <= 100: ";
    cin >> n3;
  } while (n3 < 1 || n3 > 100);


 do // loop principale di elaborazione
  {
     cout << "\n\n--- MENU DELLE OPERAZIONI ---\n";
     cout << "\nA - somma tra n1, n2 e n3.";
     cout << "\nB - prodotto tra n1 e n2.";
     cout << "\nC - sottrazione tra n3 e n1.";
     cout << "\nD - divisione tra n1 e n2.";
     cout << "\nX - uscita dal programma.";
     cout << "\n\nSelezionare un'operazione: ";
     cin >> c;

     // viene qui usato il costrutto if..else annidato nella forma
     // sintattica compatta si invita a riscrivere questa parte usando il
     // costrutto switch..case

    if (c == 'a' || c == 'A')
    {
       intres = n1 + n2 + n3;
       cout << "\nOpzione "<< c <<":" << n1 <<" + ";
        cout << n2 <<" + "<< n3 <<" = "<< intres << "\n";
     }
     else if (c == 'b' || c == 'B')
     {
       intres = n1 * n2;
       cout << "\nOpzione "<< c <<":" << n1 <<" * ";
        cout << n2 << " = "<< intres << "\n";
     }
     else if (c == 'c' || c == 'C')
```

```
      {
        intres = n3 - n1;
       cout << "\nOpzione "<< c <<":" << n3 <<" * ";
       cout << n1 << " = "<< intres << "\n";
      }
      else if (c == 'd' || c == 'D')
      {
        dblres = (float)n1 / (float)n2;
         cout << "\nOpzione "<< c <<":" << n1 <<" * ";
         cout << n2 << " = "<< dblres << "\n";
      }

  } while (c != 'x' && c != 'X');


  cout << "\n\nFINE ELABORAZIONE ----------------------------\n\n";
  system("PAUSE");
  return 0;
}
```

ESERCIZIO 5 (Costrutti iterativi + uso di contatori + uso di accumulatori + Algoritmi)

Realizzare un programma che legga una sequenza di 5 numeri interi maggiori o uguali a 1 e minori o uguali a 100 e, ad ogni nuovo numero inserito, stampi a video il valore massimo introdotto fino a quel momento. Se il numero introdotto è minore di 1 o maggiore di 100, si deve stampare un messaggio di errore e il valore massimo corrente. Al termine si stampi un messaggio di saluto, il valore massimo finale, il numero di interi validi introdotti, il numero di interi non validi introdotti.

```
#include <iostream.h>
#include <stdlib.h>
void main()
{
  int i, num, max, inrange, outrange;

  // inizializzazione variabili
  max = 0;
  inrange = 0;
  outrange = 0;

  for (i = 1; i <= 5; i++)
  {
    cout << "\nInserire un numero intero >= 1 e <= 100 (“ <<i/5 << “): ";
    cin >> num;

    if (num < 1 || num > 100) // numero fuori range
    {
       outrange++;
       cout << "\nErrore: e' stato inserito un intero fuori range!";
    }
    else  // numero OK
    {
      inrange++;
     if (num > max) max = num;  // assegna il nuovo massimo
    }

     cout << "\nIl valore massimo corrente e': “ << max << endl;
  }

  cout << "\n\nFINE ELABORAZIONE ----------------------------”;
  cout << "\nIl valore massimo finale e': “ << max;
  cout << "\nIl numero di interi validi inseriti e': “ << inrange;
  cout<<"\nIl numero di interi non validi inseriti e':”;
  cout << outrange<<“\n\n";
   system("PAUSE");
} // end main
```

ESERCIZIO 6 (Costrutti iterativi + uso di contatori + uso di accumulatori + Algoritmi)

Si scriva un programma in linguaggio C che letto un numero intero positivo dallo standard input, visualizzi a terminale il quadrato del numero stesso facendo uso soltanto di operazioni di somma.
Si osservi che il quadrato di ogni numero intero positivo N può essere costruito sommando tra loro i primi N numeri dispari.
Esempio: N = 5; $N^2 = 1 + 3 + 5 + 7 + 9 = 25$.

Stesura informale dell'algoritmo:

Premessa:
l'idea di soluzione è quella di scandire i primi N numeri dispari esprimendoli nella forma (*i + i +1*) al variare dell'intero *i* tra 0 e *N-1* e accumulare la loro somma man mano che si procede nella loro scansione in un'altra variabile.
$N^2 = (2\cdot0+1) + ... + (2\cdot i+1) + ... + (2\cdot(N-1)+1)$;
quindi si utilizzeranno almeno 3 variabili: N, i, S rispettivamente per il numero, il contatore e l' accumulatore.

1. Inizio dell'algoritmo
2. Leggi un numero intero positivo dallo standard input.
3. Inizializza un contatore i a 0
4. Inizializza un accumulatore S a 0
5. Finché il valore del contatore è minore del numero letto
    a. Somma all'accumulatore il doppio del valore del contatore incrementato di 1
    b. Incrementa il contatore
    c. Torna al punto 5.
6. Stampa a terminale il valore dell'accumulatore
7. Fine dell'algoritmo

```
#include <iostream.h> /* inclusione della libreria standard     */
#include <stdlib.h>
int main( )
{
          /* dichiarazione delle variabili                    */
  int i;  /* contatore                                        */
  int N;  /* variabile di cui si vuole calcolare il quadrato */
  int S;  /* accumulatore per il risultato del calcolo        */

  /* ciclo di controllo che garantisce N > 0                  */
  do
  {
     cout << "\n Inserisci un numero positivo N: ";
     cin >> N;         /* legge N dallo standard input */

     /* Finché il numero inserito non è positivo ripetere    */
     /* l'immissione dati.                                   */

  } while (N < =0 );


  /* Il numero inserito è positivo */

  S = 0;   /* inizializzazione della variabile di accumulo   */

  /* ciclo di scansione dei primi N numeri dispari          */
  i = 0;
  while(i < N)
  {
      /* Finché il contatore è minore del numero letto      */
      /* aggiorna il contenuto della variabile accumulatore */

      S = S + (i+i+1);
      i = i + 1;                /* incrementa il contatore */
  }

  cout << "\n Il quadrato del numero inserito e': " << S <<  "\n";

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

ESERCIZIO 7 **(Costrutti iterativi + uso di contatori + uso di accumulatori + Algoritmi)**

Si definisce *Triangolare* un numero costituito dalla somma dei primi N numeri interi positivi per un certo N.
Ad esempio: per Q = 10 si ha Q = 1 + 2 + 3 + 4, da cui N = 4.
Scrivere un programma C che stabilisca se un numero intero positivo Q, letto dallo standard input, è un numero triangolare o meno, utilizzando soltanto operazioni tra numeri interi.
In caso affermativo stampare a video il numero inserito e il massimo degli addendi che lo compongono.

Stesura informale dell'algoritmo

Idea di soluzione: se *Q – 1 – 2 – 3 – ... – i – ... – N == 0* per un certo *N* allora *Q* è Triangolare.

1. leggi il numero positivo *Q* dallo standard input
2. inizializza un contatore *i* a zero;
3. memorizza in una variabile *S* il valore della variabile in ingresso.
4. Finché il numero *S* è maggiore di zero
   - 4.1. incrementa di *1* il valore del contatore
   - 4.2. sottrai a *S* il valore del contatore *i*
   - 4.3. torna a 4.
5. Se (il valore residuo di *S* è zero) allora
   - 5.1. il numero è triangolare
   - 5.2. il valore del massimo degli addendi è uguale al contatore *i*
   - 5.3. la variabile *Q* contiene il valore della variabile in ingresso
6. altrimenti il numero NON è triangolare.

Nota: è buona norma, in generale, non modificare le variabili contenenti i dati in ingresso perché può accadere che sia necessario accedere a tali valori in diversi punti del programma. (Nel caso appena mostrato senza l'ausilio di un'altra variabile S non sarebbe possibile, al termine della computazione, stampare a video il valore del numero inserito così come richiesto dalla traccia del problema.)

```
#include <conio.h>
#include <iostream.h>
int main( )
{
         /* dichiarazione delle variabili                          */
 int i;  /* variabile contatore                                    */
 int Q;  /* variabile per il numero letto da tastiera              */
 int S;  /* variabile accumulatore                                 */

 /* ciclo di controllo per l'immissione dati che garantisce Q>0 */
 do
 {
   cout << "\n Inserisci un numero positivo Q: ";
   cin >> Q;

 }while (Q <= 0);

 S = Q;        /* copia del valore del dato in ingresso        */
 i = 0;        /* inizializzazione del contatore               */
 while (S > 0)
 {
   i = i + 1;
   S = S - i;
 }

 /* all'uscita dal ciclo il valore assunto dalla variabile S   */
 /* permette di procedere in base a due alternative            */

 if (S == 0)
 {
  /* il valore del contatore i contiene qui il valore del massimo
   degli addendi che compongono il numero triangolare inserito.*/

   cout << "\n" << Q << " = alla somma dei primi ";
   cout << i << "numeri positivi!;
 }
 else
 {
  cout<<"\n Il numero "<< Q << " non e' un numero triangolare!\n";
 }

   system("PAUSE");
  return 0;
} // end main
```