2° lezione prog II

Elementi di base di C++

Costruzione di un programma in C++

1. **Editor di testo** per scrivere il programma sorgente e salvarlo in un file chiamato file sorgente o codice sorgente. Per convenzione, i programmi scritti in C++ hanno estensione. Cpp
2. **Compilare il codice** sorgente: il compilatore traduce il codice sorgente in codice oggetto.in C++ il file prodotto in questo passo ha lo stesso nome di quello sorgente ma estensione .obj oppure .o
3. **Se il passo due ha avuto successo** il linker collega al codice oggetto le funzioni di libreria C++ di cui il programma fa uso.

**La libreria standard del C++ contiene il codice oggetto di svariate funzioni che realizzano compiti comuni come la gestione dell’ I/O o calcoli matematici**. Il collegamento produce una versione eseguibile del programma che viene posta in un file che generalmente il nome sorgente ed estensione .exe sotto Windows e nulla sotto Linux.

Immagine che contiene testo, schermata, Carattere, design

Descrizione generata automaticamente

**Messa a punto di un programma in C++**

1. **Scrittura del codice sorgente sul computer**
2. **Compilazione del codice sorgente**
3. **Collegare il codice oggetto prodotto con quello delle librerie dichiarate nel programma sorgente**
4. **Il compilatore individua eventuali errore di sintassi che devono essere corretti e quindi si torna al punto 1**
5. **Collaudo del programma. Se il programma non fa quello che il programmatore si aspettava si torna al punto u**

**Struttura generale di un programma in C++**

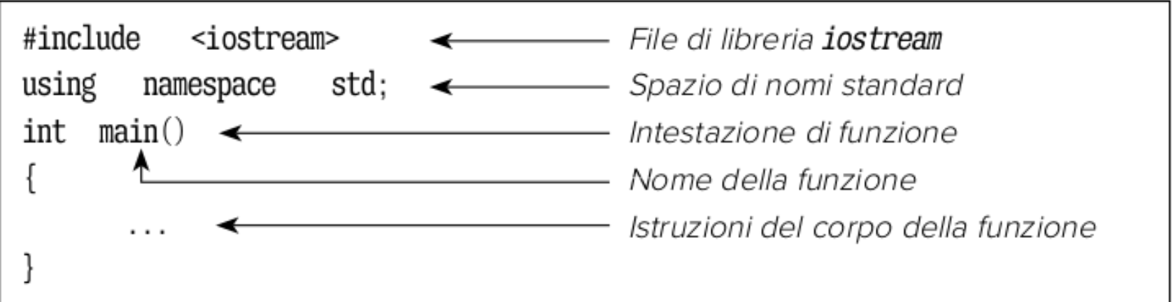


Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

Immagine che contiene testo, ricevuta, schermata, Carattere

Descrizione generata automaticamente

Immagine che contiene testo, schermata, linea, Carattere

Descrizione generata automaticamenteImmagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

La funzione int main()Immagine che contiene testo, schermata, ricevuta, Carattere

Descrizione generata automaticamente

Parole riservate del C++Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Comandi per il terminale

-   **ls**: comando che si utilizza per sapere la lista di file e di directory(cartelle) che si hanno nella directory corrente.

-   Il comando **gcc** serve per eseguire la compilazione del file c vuole come parametri il nome del file(con estensione .c ) da questo si genererà un file eseguibile che per default chiamerà a.out . per l’esecuzione ./a.out

- con – o si rinomina il file eseguibile

Fase di compilazione

1. **Fase di preprocessing:** il preprocessore legge le direttive al preprocesssore (#include). In questo modo diciamo al pre processore che esiste un file header.h  e di includere le librerie nel codice sorgente. Il pre processore legge il file heder lo interpreta (interpretando tutti gli include che ci sono all’interno di stdio.h ricorsivamente) e poi lo concatena con il nostro file sorgente e lo manda al compilatore. (gcc main.c -e con questo comando si va ad eseguire solo la fase di pre-processing)

2.   **Fase di compilazione**:  È la traduzione di tutto il pacchetto prodotto precedentemente in linguaggio macchina. Quando si va a compilare non si deve necessariamente generale eseguibile si può eseguire solamente la compilazione ma non  l’ esecuzione. Se questi file non si eseguono si hanno tante translation  unit che contengono funzioni. una di queste contiene il main e volendo si potrebbero compilare separatamente e si ottengono tanti file Object, file che contengono codice macchine ma non sono file eseguibili si arriva all'eseguibile  del programma mettendo insieme questi file object (linker). Per fare questo da terminale si utilizza il seguente comando: gcc sorgente -o.

     gcc main.c -o main.c nomina il file eseguibile. Per lanciare il programma bisogna scrivere ./main

Le variabili

Una variabile è uno spazio in memoria adeguato ad ospitare un dato di un certo tipo, è identificata da **un nome**  che è univoco. si dice che il linguaggio C è un linguaggio tipato.Ogni variabile dichiarata nel programma ha un certo indirizzo nella memoria del calcolatore.

int somma=0;

* int(intero) Si sta dichiarando una variabile di tipo  intero quindi in memoria verrà predisposto lo spazio per contenere un intero.
* somma e il nome della variabile o l'identificatore

 Inoltre questa variabile è stata inizializzata con un valore 0.

Quando si dichiara una variabile si sta dicendo al compilatore di predisporre(allocare) uno spazio in memoria per contenere un dato di un certo tipo. invece quando la si Inizializza questa variabile avrà anche un valore. È consigliabile eseguire sempre questa procedura perché le variabili potrebbero avere anche valori sporchi e quindi questo  genererebbe un errore semantico perché  da una determinato operazione  riceviamo un risultato diverso da quello che ci aspettiamo. Si possono dichiarare e inizializzare più variabili nello stesso tempo dello stesso tipo nel modo seguente: int a=2,c=3;

Per mostrare tutti i warning che ha rilevato il compilatore bisogna utilizzare il comando -Wall.

Letterali e variabili

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Tipologie di variabili

* **Int**: intero standard (**va da -2 miliardi a +2 miliardi**)
* **Unsigned int**: intero senza segno (**solo positivo**) quindi i valori che la variabile può contenere diventano 4 miliardi
* **Long**: serve per contenere numeri ancora più grandi di quelli che può contenere un int
* **long long**: serve per contenere numeri ancora più grandi del long
* **float**: numero in virgola mobile in singola precisione (3.5)
* **double**: numero in virgola mobile (ad esempio numeri con più cifre dopo la virgola) es 3.14
* **char**: serve per dichiarare che la variabile in questione dovrà contenere un carattere. Ad esempio  char c=‘c’; (in questo modo sto assegnando un carattere alla variabile c.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

**operatori di incremento/decremento**

* incremento unitario ++a (PREFISSA)
* decremento unitario --a (PREFISSA)
* decremento unitario a-- (POSTFISSA)
* incremento unitario a++ (POSTFISSA)

Immagine che contiene testo, Carattere, schermata

Descrizione generata automaticamente

Per la forma post fissa si vuole assegnare a v il valore di k e poi incrementare k.

Per la forma prefissa prima si incrementa k e poi si assegna a v

Le costanti

Sono dei dati che non cambiano nel tempo o meglio dire in  Run Time.

 Ci sono due modi:

1. **Con la direttiva al preprocessore # define** #define esempio(#define PIGRECO 3.14) all’interno del file header. In fase di compilazione tutte le occorrenze delle costanti utilizzate  vengono sostituite con il valore  letterale.
2. **Con la parola chiave const** questo metodo si utilizza quando il valore non è noto a tempo di compilazione. Per utilizzare questo metodo il valore della costante va inizializzato in fase di creazione. ES const int N=4;

**Visibilità di variabili in C++**

La zona di un programma in cui una variabile attiva si dice visibilità o scope della variabile. Le variabili C++ possono essere:

* + **Locali**: sono quelle definite all’interno di un blocco di istruzioni, tipicamente quello di una funzione, e sono visibili dal punto in cui sono state definite fino alla fine di quel blocco
  + **Globali**: sono quelle che si dichiarano al di fuori di qualunque blocco, quindi anche al di fuori del funzione main(), e sono visibili dappertutto quindi da qualunque funzione, meno che nei blocchi in cui esistono variabili locali con lo stesso nome

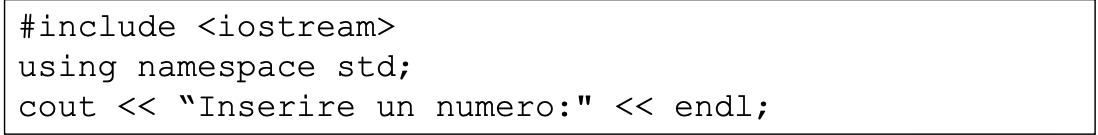
**Input e output da control**

Immagine che contiene Elettrodomestico, design, elettrodomestico

Descrizione generata automaticamente

**Il linguaggio C++ gestisce le operazioni di IO mediante il concetto di flusso o stream. Un flusso è una astrazione utile a modellare le comunicazioni tra le applicazioni cioè i processi e l’ambiente (il sistema operativo e hardware).**

**In C++ le librerie per gli stream sono implementate mediante gerarchie di classi che permettono di indipendenza del codice per la gestione dell’operazione di IO dallo specifico dispositivo.** **Gli stream si possono collegare di volta in volta entità specifiche come la tastiera e allo schermo eccetera eccetera**.

**Input e output da console**

Lo standard output predefinito è generalmente associato al video (oggetto count).

L’operatore << e l’operatore di inserimento che spinge i dati alla sua destra nel canale di output cout che di tipo ostream. Invece l’input generalmente associata alla tastiera ed ha un oggetto predefinito std::cin. L’operatore di Estrazione >> viene usato per leggere i dati dando standard in input e direzione al diversi variabile X in questo caso.  
 Immagine che contiene testo, schermata, Carattere, ricevuta

Descrizione generata automaticamente

**cout,cin,cerr,** sono oggetti che danno scope globale**. La clausola using namespace std**; **permette di risolvere i nomi (classi, funzioni, costanti)** namespace std; **senza dover usare l’operatore risolutore di scope (::)**Immagine che contiene testo, Carattere, schermata, bianco

Descrizione generata automaticamente

**In questo caso se l’utente non dovesse inserire un numero nella variabile non sarà copiata ad alcun valore e verrà generato un errore di IO. In questo caso il metodo fail() di cin restituirà true e l’input buffer verrà mantenuto in uno stato di errore. Per questo occorre chiamare cin.claer() per svuotare il buffer per permettere i i vari input successivi.** **cin.inglore() serve ad indicare che vogliamo indicare l’input errato acquisito fino alla generazione dell’errore. Cin.clear() pone a zero tutti i Flag errore dello stream. Non si può procedere molta operazione di IO senza aver prima posto a zero i flags.** **cin.inglore() si utilizza per scartare i caratteri rimasti nello stream.**

Se bisogna acquisire una stringa è consigliabile utilizzare la funzione **getline().**

**Le stringhe**

**Una stringa è una sequenza di caratteri che termina con il carattere speciale ‘\0’ che rappresenta la sua fine**. In C++ si ha a disposizione la classe stringa che permette di manipolare le stringhe agevolmente.

Immagine che contiene testo, schermata, Carattere, bianco

Descrizione generata automaticamente

**Tra le varie funzioni che offre la classe e string vi sono lenght() che restituisce il numero di caratteri di una stringa. substr che estrae una sotto stringa** e **restituisce un oggetto stringa vuole come parametro l’indice del primo carattere e la lunghezza della sotto stringa.** Se il secondo parametro non viene specificato saranno considerati tutti i rimanenti caratteri fino alla fine e la stringa originale.

**Le principali funzioni della libreria <cstring>**

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamenteImmagine che contiene testo, schermata, ricevuta, Carattere

Descrizione generata automaticamente

**Conversione di caratteri a numeri ed i numeri caratteri**

Molto spesso bisogna convertire stringhe di cifre nei rispettivi formati numerici. C++ fornisce per questi scopi le funzioni aatoi atof ed atol.

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

**Numeri pseudocasuali**

Sono applicati in diversi ambiti come per la gestione di magazzini o in ambito scientifico/finanziario. I numeri devono essere randomici, il generatori di numeri casuali deve essere controllabile cioè che con lo stesso input deve essere in grado di generare lo stesso (seme) risultato, deve avere la portabilità (su diverse architetture), ed efficiente intesa come le risorse di calcolo che occupa la propria esecuzione.

Le funzioni rand() ed srand() di cstdlib

**Per utilizzare queste funzioni bisogna utilizzare l’header cstdlib. e l’header ctime per utilizzare la funzione time().**

* **rand()**: restituisce un numero intero e non prende parametri in ingresso. Alla chiamata restituisce il prossimo numero della sequenza. Il numero restituito e’ tra 0 e RAND\_MAX.
* **srand()**:**ha come paramento in ingresso(seme o seed).** Si utilizza quando si vuole rei inizializzare la sequenza con un certo numero chiamato seed. Non si conosce l’algoritmo che genera numeri casuali ma ci basti sapere che a semi differenti corrispondono sequenze randomiche differenti. Se non si invoca la funzione srand() per default il seme sarà 1. Se si inserisci lo stesso seme si otterrà la stessa sequenza di numeri casuali. Per default. Specificare il seme è come inizializzare la sequenza. La funzione time() restituisce il tempo in secondi percorsi dal 1° gennaio del 1970 e viene utilizzata quando ad esecuzioni della funzione rand() devono restituire sequenze di numeri diverse, in questo modo: srand(time(0)); oppure srand(time(NULL));

**Come generare numeri casuali da 0 a N-1**

Si procede nel seguente modo: **rand()%N (DA 0 A N-1).**

**Come generare numeri casuali in un range [a, b]**

Dati due numeri interi a e b con a<b **rand((b-a+1)+a))**

* Con **b-a+1** si ottiene l’ampiezza dell’intervallo
* Per farlo partire da a e non da 0 si procede nel seguente modo: **((b-a+1)+a))**

**Come generare numeri casuali in virgola mobile**

Si procede nel seguente modo**: doble n=rand()/(RAND\_MAX\*1.0);**

**Come generare numeri casuali a partire da un range di due numeri in virgola  mobile**

Si prende questo numero compreso tra 0 e 1 si riscala in modo da ottenere un numero compreso tra 0 e Y- X e poi aggiungere X. in codice: **double r=((rand()/RAND\_MAX\*1.0)\*x-y)+x**

* **Numero compreso tra 0 e 1**
* **Numero compreso tra y - x**
* **Numero compreso tra X e Y( aggiungendo X a tutta l'espressione.).**