Ingresso e Uscita
Variabili e costanti intere

Ingresso e Uscita in C++

- Il linguaggio C++ non prevede istruzioni per l'ingresso/uscita
- Implementato mediante *oggetti di libreria* chiamati *stream*
  - *stream: flusso di caratteri*
  - *ostream*: flusso di caratteri in uscita, output formattato
  - *istream*: flusso di caratteri in ingresso, input formattato

Flussi

- Flusso di caratteri:
  - successione di righe,
  - ciascuna costituita da zero o più caratteri, e
  - terminata dal carattere speciale newline '*n*'

cin, cout, cerr 1/2

- Quando un programma inizia la propria esecuzione ci sono tre flussi di caratteri già aperti
  - *cin*: flusso standard di ingresso (standard input)
  - *cout*: flusso standard di uscita (standard output)
  - *cerr*: flusso standard di uscita per comunicare messaggi di errore

cin, cout, cerr 2/2

- Se il programma è invocato da una shell Unix senza *redirezionamenti*
  - Lettura da *cin*:
    - Lettura dei caratteri immessi dal terminale in cui gira la shell (tastiera)
  - Scrittura su *cout* o su *cerr*:
    - Visualizzazione sul terminale in cui gira la shell

Operatore di uscita <<

- Scrittura formattata su *cout*
- *cout<<obj1<<obj2<<...<<endl ;*

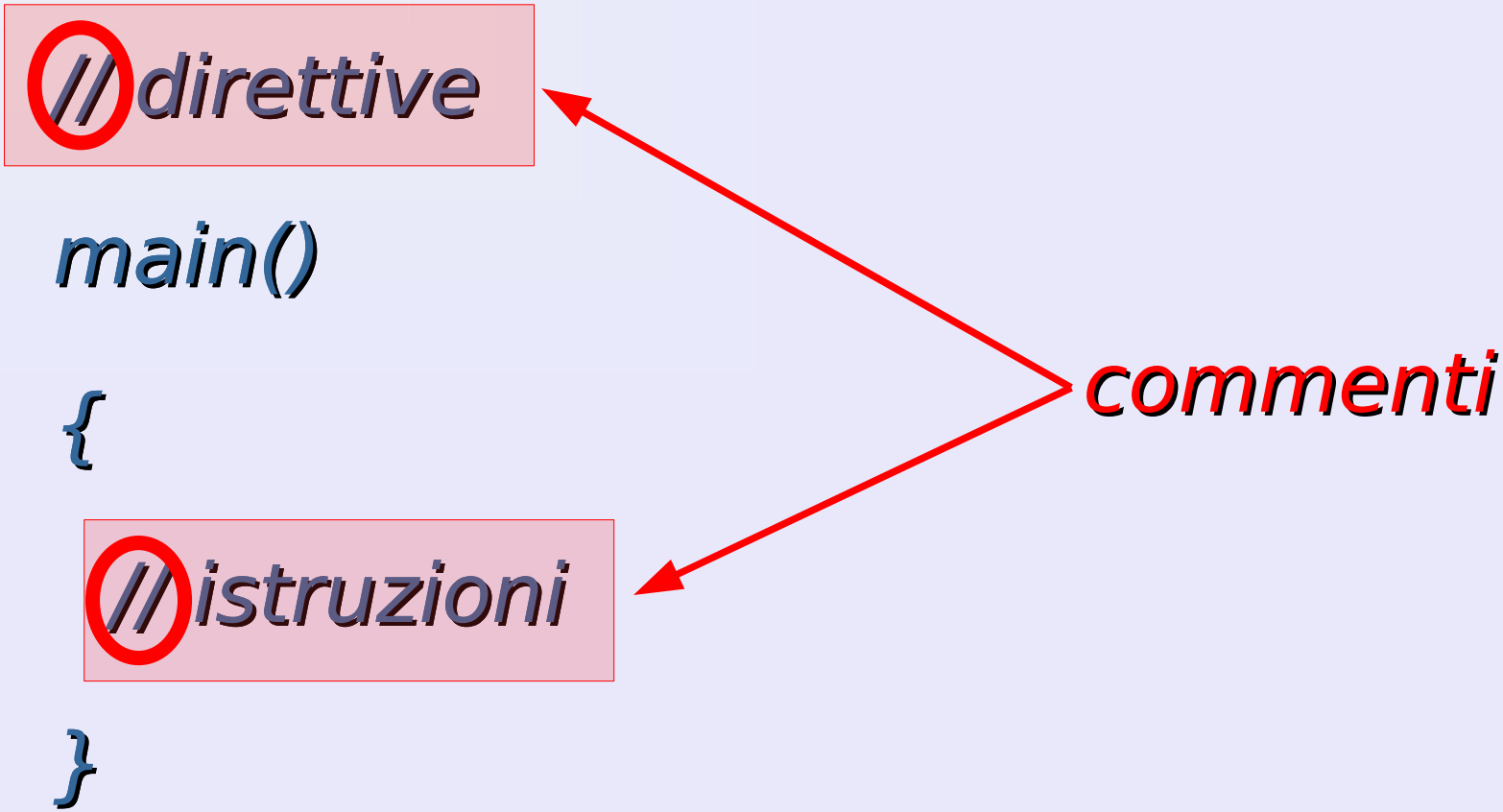
iostream

- Ogni file sorgente che contenga riferimenti ad oggetti della libreria di ingresso/uscita deve contenere le direttive

  #include <iostream>
  using namespace std;

  - Devono precedere il primo punto in cui viene utilizzato uno stream di ingresso/uscita

Anatomia programma C++



Primo esercizio 4/4

```
#include <iostream>

using namespace std;

main()

{

  cout<<"Ciao mondo!\n" ;

}
```

Esercizio 2 2/2

```
#include <iostream>

using namespace std;

main()

{

  cout<<"Ciao mondo!"<<endl ;

}
```

Compilazione

- Sintassi più semplice per generare un programma eseguibile da un file sorgente:
- *g++ nome_sorgente.cc*
  - Assegna un nome predefinito al programma eseguibile (a.out)
- *g++ -o nome_eseguibile nome_sorgente.cc*
  - Permette di scegliere il nome del programma eseguibile

Messaggi di errore 2/2

- Se ci sono problemi, il compilatore può comunicare
  - Warnings (avvisi): c'è qualcosa di 'sospetto' nel codice, ma si può comunque generare un eseguibile
  - Errors: ci sono errori che impediscono la conclusione della compilazione
- **LEGGETELI per capire cosa c'è che non va nel programma !!!**

Esercizio 2 1/2

- Scrivere un programma in cui si definisce una costante intera e se ne stampa il valore sullo schermo

Esercizio 2 1/2

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

main()

{

  const int i = 10 ;

  cout<<i;

}
```

Esercizio 3 1/2

- Scrivere un programma in cui si definisce una costante intera e se ne stampa il valore sullo schermo col seguente formato:

Il valore della variabile è 10.

- E si va a capo

Esercizio 3 2/2

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

main()

{

  const int i = 10 ;

  cout<<"Il valore della variabile è
   “<<i<<”.”<<endl ;

}
```

Una soluzione alternativa

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

main()

{

  const int i = 10 ;

  cout<<"Il valore della variabile è "<<i ;

  cout<<"."<<endl ;

}
```

Ritorno valore finale

- Un processo ritorna un valore quando termina
- Può essere letto ed utilizzato dal programma che lo ha invocato
- La funzione *main* può essere dichiarata ritornare un valore di tipo intero
- Il valore di ritorno della funzione main è il valore ritornato dal processo stesso

Esempio

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

int main()

{

  const int i = 10 ;

  cout<<”Il valore della variabile è "<<i ;

  cout<<”.”<<endl;

  return 0 ;

}
```

Lettura dallo stdin 1/2

- Operatore di ingresso >> applicato ad un oggetto di tipo *istream*
- *Esempio: cin>>nome_variabile ;*
- Legge i caratteri in ingresso dallo standard input
- Li interpreta in base al tipo della variabile
- Assegna il valore letto alla variabile di nome *nome_variabile*

Esempio di interpretazione

- *cin>>a;*
- La variabile *a* è di tipo int
- Si leggono i caratteri 2, 3 e *\n*
- Vengono interpretato come le due cifre decimali del numero 23
- Il numero 23 viene memorizzato nella variabile *a*

Esercizio 4 1/2

- Si scriva un programma che legge un valore intero da tastiera e lo stampa a video

Esercizio 4 2/2

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

main()

{

  int i ;

  cin>>i ;

  cout<<"Il valore inserito è "<<i<<endl ;

}
```

Ingresso inconsistente

- Cosa accade se la sequenza di caratteri letta non rappresenta alcun numero in notazione decimale?
- Il valore del secondo argomento rimane invariato (non avviene alcuna memorizzazione)
- Le successive letture falliranno

Esercizio 5 1/2

- Miglioriamo l'esercizio precedente
- Vogliamo stampare anche un messaggio di richiesta del numero da inserire:

Inserisci un valore intero: 13

Il valore inserito è: 13

Nota

- L'operatore di ingresso >> applicato al *cin* **non scrive sullo standard output**

Esercizio 5 2/2

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

main()

{

  int i ;

  cout<<”Inserisci un valore intero " ;

  cin>>i ;

  cout<<"Il valore inserito è “<<i<<endl ;

}
```

Esercizio 6 1/3

- Scrivere un programma che legge in ingresso due valori interi e stampa il risultato della moltiplicazione tra i due numeri

Inserisci il primo numero: 10

Inserisci il secondo numero: 20

*10 * 20 = 200*

Esercizio 6 2/3

```
#include <iostream>
using namespace std;
main()
{
  int i, j, ris ;

  cout<<"Inserisci il primo numero " ;
  cin>>i ;
  cout<<"Inserisci il secondo numero " ;
  cin>>j ;

  ris = i * j ;
  cout<<i<<"*"<<j<<" = "<<ris<<endl ;
}
```

Esercizio 6 3/3

```
#include <iostream>   /* Soluzione alternativa: */
using namespace std;  /* senza variabile di     */
main()                /* appoggio               */
{
  int i, j ;

  cout<<"Inserisci il primo numero " ;
  cin>>i ;
  cout<<"Inserisci il secondo numero " ;
  cin>>j ;

  cout<<i<<"*"<<j<<" = "<<i*j<<endl ;
}
```

Esercizio 7 1/3

- Scrivere un programma che legge in ingresso due valori interi e stampa sia il risultato della divisione intera tra i due numeri che il resto della divisione stessa

  Inserisci il primo numero: 5

  Inserisci il secondo numero: 2

  5 / 2 = 2 con resto 1

Esercizio 7 2/3

```
#include <iostream>
using namespace std;
main()
{
  int i, j, div, resto ;

  cout<<"Inserisci il primo numero " ;
  cin>>i ;
  cout<<"Inserisci il secondo numero " ;
  cin>>j ;

  div = i / j ;
  resto = i % j ;
  cout<<i<<" / "<<j<<" = "<<div<<" con resto
    "<<resto<<endl ;
}
```

Esercizio 7 3/3

```
#include <iostream>      /* Soluzione alternativa      */
using namespace std;     /* senza variabili di         */
main()                   /* appoggio                   */
{
  int i, j ;

  cout<<"Inserisci il primo numero " ;
  cin>>i ;
  cout<<"Inserisci il secondo numero " ;
  cin>>j ;

  cout<<i<<" / "<<j<<" = "<<i/j<<" con resto
    "<<i%j<<endl ;
}
```

Esercizio 8 1/4

- Scrivere un programma che legge in ingresso due valori interi e li memorizza in due variabili, quindi scambia il contenuto delle variabili e lo stampa sullo schermo

Inserisci il valore di i: 2

Inserisci il valore di j: 3

Dopo lo scambio: i = 3, j = 2

Esercizio 8 2/4

- Pensare prima all'algoritmo
- Forse ci serve una variabile d'appoggio ...

Esercizio 8 3/4

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

int main()
{
  int i, j ;

  cout<<"Inserisci il valore di i " ;
  cin>>i ;
  cout<<”Inserisci il valore di j " ;
  cin>>j ;
```

Esercizio 8 4/4

```
  Int appoggio = i ;
  i = j ;
  j = appoggio ;

  cout<<"Dopo lo scambio: i = "<<i
        <<", j = "<<j ;
  return 0 ;
}
```

Esercizio 9

- Scrivere un programma che legge in ingresso due valori interi e li memorizza in due variabili, quindi scambia il contenuto delle variabili e lo stampa sullo schermo
- Ma senza utilizzare nessuna variabile d'appoggio!
- *scambia_senza_appoggio.cc*

Altri dettagli sui processi

- Ad ogni processo è associato un identificatore numerico: *pid*
- Per elencare i propri processi:

ps x

- Per elencare tutti i processi:

ps ax

Ancora sui segnali

- Ctrl + C manda il segnale SIGINTR (interruzione) al processo in esecuzione
- A volte tale segnale può non bastare per interrompere il processo
- Comando *kill*: spedisce segnali ai processi

  kill -signal pid

- Eventualmente invocato da un altro terminale

Compiti per casa 1/9

- Scrivere i seguenti programmi
  - **senza** utilizzare istruzioni di **controllo di flusso** (niente istruzioni condizionali ed iterative, ma solo esecuzione sequenziale)
  - facendo uso **solo** di variabili di tipo *int* e dei relativi operatori
  - +, -, *, /, %, ++, *abs()*

Compiti per casa 2/9

- Scrivere un programma che legge in ingresso un numero intero, lo interpreta come un tempo espresso in secondi, e lo stampa in minuti e secondi (*da_sec_a_min_sec.cc*)

Tempo in secondi? 67

Equivalgono a 1 min, 7 sec

Compiti per casa 3/9

- Scrivere un programma che legge in ingresso due numeri, li interpreta come un tempo espresso in minuti e secondi, e lo stampa in secondi (*da_min_sec_a_sec.cc*)
  - Attenzione: assumiamo come valido anche un ingresso in cui il secondo numero sia maggiore di 59

Minuti ? 3

Secondi ? 78

Equivalgono a 258 secondi

Compiti per casa 4/9

- Scrivere un programma che legge in ingresso un numero intero e stampa 0 se il numero è pari, 1 altrimenti (*0_se_pari.cc*)

  Inserisci un numero intero: 23

  1

- Scrivere un programma che legge in ingresso un numero intero e stampa 1 se il numero è pari, 0 altrimenti (*1_se_pari.cc*)

Compiti per casa 5/9

- Scrivere un programma che legge in ingresso due numeri interi positivi, poi stampa 0 se il primo è multiplo dell'altro, 1 altrimenti (*0_se_multiplo.cc*)

Inserisci il primo numero intero positivo: 32

Inserisci il secondo numero intero positivo: 11

1

Compiti per casa 6/9

- Scrivere un programma che legge in ingresso due numeri interi positivi, poi stampa 1 se il primo è multiplo dell'altro, 0 altrimenti (*1_se_multiplo.cc*)

Inserisci il primo numero intero positivo: 32

Inserisci il secondo numero intero positivo: 11

0

Compiti per casa 7/9

- Scrivere un programma che legge in ingresso un numero intero diverso da 0, e stampa -1 se è negativo, 1 se è positivo (*1_se_pos-1_se_neg.cc*)

  Inserisci un numero intero: -3

  -1

Compiti per casa 8/9

- Scrivere un programma che legge in ingresso un numero intero diverso da 0, e stampa 0 se è negativo, 1 se è positivo (*0_se_neg_1_se_pos.cc*)

  Inserisci un numero intero: -3

  0

Compiti per casa 9/9

- Scrivere un programma che legge in ingresso un numero intero diverso da 0, e stampa 1 se è negativo, 0 se è positivo

  Inserisci un numero intero: -3

  1