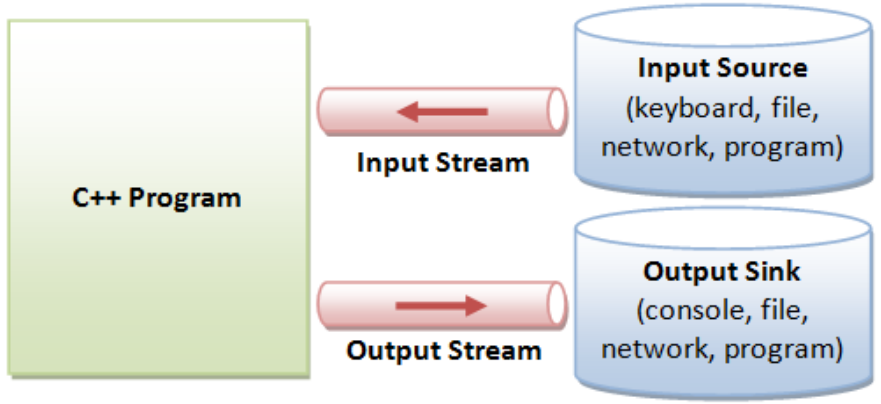
UNIVERSITÀ
DI PARMA

C++ input/output

Alberto Ferrari

- l'***input/output*** del C++ è basato sugli ***stream***
- gli stream sono ***sequenze di byte*** che rappresentano il ***flusso*** in entrata o in uscita di un programma
- gli stream fungono da ***intermediari*** fra i programmi e le periferiche di I/O liberando il programmatore dalla necessità di gestire direttamente le periferiche



Internal Data Formats:
- Text: `char`, `wchar_t`
- `int`, `float`, `double`, etc.

External Data Formats:
- Text in various encodings (US-ASCII, ISO-8859-1, UCS-2, UTF-8, UTF-16, UTF-16BE, UTF16-LE, etc.)
- Binary (raw bytes)

stream di input

- nelle operazioni di input i dati vanno da una ***sorgente di input*** verso il ***programma***
  - una sorgente di input può essere
    - la ***tastiera*** *(console input standard)*
    - un ***file***
    - una ***risorsa di rete***
    - l'***output*** di un altro ***programma***

stream di output

- nelle operazioni di output il flusso di dati che ha come ***sorgente*** il ***programma*** può essere diretto verso:
  - il ***video*** (*console output standard*)
  - un ***file***
  - la ***rete***
  - un altro ***programma***

- **<<** e **>>** sono ***operatori*** di flusso che operano sugli ***stream***
- << ***estrazione*** da uno stream
- >> ***inserimento*** in uno stream
- ***cin***, ***cout*** (e ***cerr***) sono oggetti che rappresentano stream di input, output ed error standard
  - ***c***onsole ***in***put
  - ***c***onsole ***out***put
  - ***c***onsole ***err***or

I/O console

cin >> cout <<

rapprOttaleEsadecimale.cpp

```
using namespace std;
int main()
{
	int n;
	cout << "Inserisci un valore intero in rappresentazione base 10: ";
	cin >> dec >> n;
	cout << dec << "rappresentazione decimale: " << n << " esadecimale: " << hex << n <<endl;
	cout << dec << "rappresentazione decimale: " << n << " ottale     : " << oct << n <<endl;
	cout << "Inserisci un valore intero in rappresentazione base 16: ";
	cin >> hex >> n;
	cout << dec << "rappresentazione decimale: " << n << " esadecimale: " << hex << n <<endl;
	cout << dec << "rappresentazione decimale: " << n << " ottale     : " << oct << n <<endl;
	cout << "Inserisci un valore intero in rappresentazione base 8: ";
	cin >> oct >> n;
	cout << dec << "rappresentazione decimale: " << n << " esadecimale: " << hex << n <<endl;
	cout << dec << "rappresentazione decimale: " << n << " ottale     : " << oct << n <<endl;
	cout << "Attenzione l'ultima base impostata rimane attiva!" << endl;
	cout << "Inserisci un valore intero : ";
	cin >> oct >> n;
	cout << dec << "rappresentazione decimale: " << n << " esadecimale: " << hex << n <<endl;
	cout << dec << "rappresentazione decimale: " << n << " ottale     : " << oct << n <<endl;
	return 0;
}
```

```
Inserisci un valore intero in rappresentazione base 10: 12
rappresentazione decimale: 12 esadecimale: c
rappresentazione decimale: 12 ottale     : 14
Inserisci un valore intero in rappresentazione base 16: 12
rappresentazione decimale: 18 esadecimale: 12
rappresentazione decimale: 18 ottale     : 22
Inserisci un valore intero in rappresentazione base 8: 12
rappresentazione decimale: 10 esadecimale: a
rappresentazione decimale: 10 ottale     : 12
Attenzione l'ultima base impostata rimane attiva!
Inserisci un valore intero : 12
rappresentazione decimale: 10 esadecimale: a
rappresentazione decimale: 10 ottale     : 12
```

I/O di stringhe

- l'operatore **<<** consente di ***inserire*** un oggetto string in un ***output*** stream
- es:
```
string message("Hello");
cout << message;
```
- l'operatore **>>** consente di ***estrarre*** un oggetto string da un ***input*** stream
- ignora eventuali caratteri di spaziatura iniziali, la lettura si ferma al primo carattere di ***spaziatura***
- es:
```
string s1, s2;
cin >> s1 >> s2;
```
- lettura di un'***intera linea*** di input con la funzione `getline(aStream, aString)` definita nella libreria string

I/O file

fstrem

- insieme di dati memorizzati su un supporto di memoria non volatile
- globale ai programmi
  - un file può essere scritto da un programma e letto da un programma diverso scritto anche in un altro linguaggio di programmazione
- la gestione concreta dei file è demandata al file system del sistema operativo
- il file system fornisce all'utente una versione astratta dell'organizzazione dei file
  - esistono diversi tipi di file system
  - tutti basati sul concetto di struttura gerarchica di cartelle (directory)
    - le cartelle vengono viste come file speciali

operazioni sui file

- ***apertura***
  - necessaria per le successive operazioni di lettura e scrittura
  - controllo dei diritti di accesso e meccanismi di gestione della concorrenza
  - viene creato un buffer in memoria per la gestione dei dati in transito tra il programma e il file
- ***chiusura***
  - non verranno più effettuate operazioni di lettura e scrittura
  - rilascio del buffer
  - operazione necessaria per la gestione della concorrenza
- ***lettura***
  - dati trasferiti dal file nel buffer
  - gestione dei dati da parte del programma
- ***scrittura***
  - dati memorizzati nel buffer (temporaneamente)
  - file system gestisce la scrittura fisica sul file

file di testo in C++

- ***file sequenziali***
  - accesso ai dati nello stesso ***ordine*** in cui sono stati inseriti
- ***ofstream*** (*output file stream*)
  - ***output*** su memoria di massa
  - file con accesso in ***sola scrittura***
- ***ifstream*** (*input file stream*)
  - ***input*** da memoria di massa
  - file con accesso in ***sola lettura***
- ***fstream*** (*file stream*)
  - ***input e output*** su memoria di massa
  - file con accesso in ***lettura e/o in scrittura***

es. scrittura file di testo

```
#include <iostream>
#include <fstream>

int main() {
   std::ofstream mioFile;
   mioFile.open("pasw03c02.txt");
   mioFile << "first line" << std::endl;
   mioFile << "second line" << std::endl;
   mioFile.close();
   return 0;
}
```

- miofile (oggetto ***ofstream***) è il ***nome logico*** del file
  - nome che lo identifica all'interno del programma
- ***open*** (***apertura*** del file e collegamento fra il nome logico e il nome fisico
  - per ofstream il file viene in ogni caso ***creato*** e ***riscritto*** anche se già presente
- pasw03c02.txt è il ***nome fisico*** (nome del file su disco (***pathname***)
- **<<** inserimento in stream (***scrittura***)
- ***close*** (chiusura)
  - viene eliminato il buffer ed effettuata la scrittura fisica

es. lettura di stringhe da file di testo

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main() {
   string s;
   ifstream mioFile;
   mioFile.open("pasw03c02.txt");
   if (mioFile.fail()) {
     cout << "errore file" << endl;  return 1;
   }
   mioFile >> s;
   while(!mioFile.eof()) {
     cout << s << endl;
     mioFile >> s;
   }
   mioFile.close();
   return 0;
}
```

- ***ifstream*** (input file stream)
- ***open*** apertura e associazione ***nome logico - nome fisico***
- il file deve esistere (non viene creato) ***fail*** verifica se l'apertura ha avuto successo
- **>>** estrazione (*stringa*) dallo stream
  - il separatore può essere ***spazio*** o ***endline***
- l'operazione di lettura può rivelare che il file è terminato
- ***eof*** (***end of file***) vale true se si è raggiunta la ***fine del file***
- ***close*** (chiusura file)

es. lettura e scrittura

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
   fstream f1,f2;
   int r; char p,c;
   f1.open("pasw03c05.txt", ios::out);
   f1 << 'k' << 'e' << 1 << endl;
   f1 << 'Q' << 'd' << 8 << endl;
   f1.close();
   f2.open("pasw03c05.txt", ios::in);
   f2 >> p;
   while(!f2.eof()) {
     cout << p << "(";
     f2 >> c;
     cout << c << ",";
     f2 >> r;
     cout << r << ")" << endl;
     f2 >> p;
   }
   f2.close();
}
```

- ***fstream*** (file stream)
- ***open***
  - ***ios::out*** (output)
    - il file viene creato
  - ***ios::in*** (input)
    - controllare fail
  - ***ios::app*** (append)
    - il file deve esistere
    - scritture in aggiunta al file precedente

es. uso dei separatori

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
int main() {
     fstream f1,f2;
     int i; char c; string s; double d;
     f1.open("pasw03c06.txt", ios::out);
     f1 << 123 << " " << 'a' << '\t' << "text"
        << " " << 3.14  << endl;
     f1.close();
     f2.open("pasw03c06.txt", ios::in);
     f2 >> i >> c >> s >> d;
     cout << "i: " << i << endl;
     cout << "c: " << c << endl;
     cout << "s: " << s << endl;
     cout << "d: " << d << endl;

     f2.close();
     return 0;
}
```

- ***separatori*** fra i valori memorizzati nel file
  - ***" "*** *(blank)*
  - ***'\t'*** (tabulazione)
  - ***endl*** (CRLF fine riga)

lettura righe

```
ifstream cost{"costituzione.txt"};
string riga1,riga2;
cout << "- prime due righe (non legge endl) -"
     << endl;
if (cost.good()) {
   getline(cost, riga1);
   getline(cost, riga2);
   cout << riga1 << riga2 << endl;
}
cost.close();

cout << "--- tutto il testo (fino a eof) ---"
     << endl;
cost.open("costituzione.txt");
getline(cost, tuttoIlTesto, '\0');
cout << tuttoIlTesto << endl;
cost.close();
```

```
cout << "--- tutto il testo (riga per riga) ---"
     << endl;
cost.open("costituzione.txt");
while (getline(cost, riga)) {
   cout << riga << endl;
}
cost.close();
```