Fondamenti di Informatica A
Laboratorio 7
Rudimenti di C

Danilo Pianini
danilo.pianini@unibo.it
Mirko Viroli
mirko.viroli@unibo.it

Alma Mater Studiorum—Università di Bologna

24 aprile 2013

Operazioni preliminari

- Accendere il computer
- Effettuare il log-in con le proprie credenziali istituzionali
  - Normalmente sono simili a:
    `nome.cognome@studio.unibo.it`
- Aprire Google Chrome, o altro browser
- Collegarsi al sito `http://bit.ly/finfa2013-e7`
- Scaricare il pacchetto `http://bit.ly/finfa2013-e7-code`
- Scompattare il pacchetto in una cartella a piacimento: questa sarà la vostra directory di lavoro di oggi.
- All'interno del pacchetto, troverete degli esempi di codice del prof. Viroli. Prendetevi del tempo per osservarli, cercate di capire cosa fanno, ed eventualmente utilizzateli come ispirazione per svolgere gli esercizi.

compilazione di un file C

- Utilizzando il comando `gcc` visto a lezione, si compili il file `compile_me.c`
- Si consiglia di usare sempre l'opzione `-Wall`, per ricevere tutti i warning generati dal vostro programma.
- Utilizzando l'opzione `-o` del compilatore, si produca l'eseguibile `UselessProgram`
- Si controlli il sorgente del programma prevedendo il suo funzionamento, e poi lo si esegua nel seguente modo:
  - Sotto Windows, scrivendo una delle seguenti:
    - UselessProgram.exe
    - UselessProgram
    - .\UselessProgram
    - .\UselessProgram.exe
  - Sotto UNIX (MacOS, Linux), scrivendo:
    - ./UselessProgram

La tua nuova migliore amica: `void printf()`

- Si crei un nuovo file `helloworld.c`, il cui `main()` utilizzi la funzione `printf` per stampare a video la stringa "Hello, world!", quindi vada a capo.
  - Si compili con il nome di `HelloWorld` e si esegua.
- Si crei un nuovo file `evens.c`, il cui `main()` utilizzi la funzione `printf` per stampare a video i primi 500 numeri pari separati da uno spazio, quindi vada a capo.
  - Si compili con il nome di `Evens` e si esegua.

Nota: per utilizzare la funzione `printf`, è necessario includere i prototipi delle funzioni fornite dalla libreria di sistema `stdio`. Per farlo, utilizzare la direttiva `#include <stdio.h>`

La tua prima funzione in C

Si costruisca in C la funzione con prototipo `int mcm(int, int)`, che dati in ingresso due interi torna il loro minimo comune multiplo. Si costruisca una funzione `int test(void)` che ne verifichi il funzionamento, e si usi il main per stampare a video il risultato di tale test. Si compili come `mcm` e si esegua.

- Si noti che, dati due numeri a e b, il loro mcm si ottiene, ad esempio, sommando al più piccolo dei due, ad ogni iterazione, il suo valore iniziale, fin quando i due numeri sono uguali.
- Si costruisca mcm in modo che depositi subito a e b in due nuove variabili a0 e b0. A questo punto si esegua un for che ad ogni passo aggiunge al più piccolo fra a e b il corrispondente a0 o b0. Tale for terminerà quando a e b sono uguali fra loro, e a quel punto si ritornerà o a o b.
- Esempio di mcm(2,3):
  1. a = 2, b = 3, a0 = 2, b0 = 3;
  2. a < b $\rightarrow$ a = 4, b = 3, a0 = 2, b0 = 3;
  3. a > b $\rightarrow$ a = 4, b = 6, a0 = 2, b0 = 3;
  4. a < b $\rightarrow$ a = 6, b = 6, a0 = 2, b0 = 3;
  5. a == b $\rightarrow$ si ritorna 6;

La tua seconda funzione in C

Con riferimento alla funzione che realizza il fattoriale vista a lezione:

- Si modifichi la funzione in maniera tale che utilizzi il tipo di dato `unsigned long int`
- Si definisca tramite macro `MAX_VAL`, che rappresenta il più grande valore che, passato in ingresso alla funzione, porta a computare un valore corretto
  - Per trovare il valore, si potrebbe ad esempio iniziare a stampare diversi fattoriali, vedendo per quale valore per primo si ha overflow.
- Si modifichi la funzione in maniera tale che, qualora il valore passato come parametro sia maggiore di `MAX_VAL`, essa ritorni 0
- Si scriva una appropriata funzione `int test(void)` che ne verifica il funzionamento
- Si faccia in modo che il main esegua il test e stampi a video "`OK`" se il test passa, "`ERROR`" altrimenti.

Scrittura di funzioni matematiche: `math.h`

La libreria math.h contiene una serie di funzioni matematiche pronte all'uso, si veda `http://www.cplusplus.com/reference/cmath/`. La si usi per costruire all'interno di un file `functions.c` le seguenti funzioni:

- `double ex(double x)` — calcola e^x
- `double sinx(double x)` — calcola $sin(x)$
- `double tanx(double x)` — calcola $tan(x)$
- `double logx(double x)` — calcola $log(x)$

Queste funzioni non devono far altro che chiamare quelle di `math.h` al loro interno.

Creazione di una libreria di funzioni

Si implementino inoltre le seguenti:

- `int minThree(int a, int b, int c)` — torna il più piccolo di tre interi
- `int randInInterval(int a, int b)` — crea un numero casuale compreso fra `a` e `b`. Si utilizzi la funzione `int rand(void)`, fornita in stdlib.h (si veda `http://www.cplusplus.com/reference/cstdlib/`)
- `int previousSquare(int x)` — calcola il più grande quadrato perfetto minore o uguale ad `x`
- `double solveEquation(double a, double b, double c)` — funzione che calcola una soluzione dell'equazione $ax^2 + bx + c$, tornando NaN se non vi sono soluzioni
  - Si noti che le soluzioni di tale equazione sono ovviamente $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
  - Per tornare NaN, si può utilizzare la macro NAN definita in `math.h`

Scrittura di un header file

Si crei un header file `functions.h` contenente tutti i prototipi delle funzioni realizzate in precedenza. Come riferimento, si consideri il file `filters.h` allegato al codice.
Si includa l'header appena creato in `functions.c`, utilizzando la direttiva:
`#include "functions.h"`
Si compili il file in maniera tale da ottenere una libreria (opzione `-c`). Si ricordi di utilizzare la flag `-lm`, che indica al compilatore di caricare la libreria `math.h`.

Uso della libreria creata

- Si crei un nuovo file `uselib.c`
- Si includa il file header appena creato
- Si scriva in uselib.c una funzione `int main()` che utilizzi al suo interno tutte le funzioni della libreria precedente, stampando a video i risultati delle prove che esegue.
- Si compili, ricordandosi di includere `functions.o` fra i file di libreria, e si esegua il programma.
- Dato che questo programma utilizza la libreria `functions.h`, che a sua volta utilizza `math.h`, è necessario utilizzare la flag `-lm` quando si compila!

Uso di `void printf()`

Si realizzi quanto segue:

- Un programma, `zeromatrix.c`, che stampi a video una matrice 30×30 di zeri. Per farlo, si realizzi una funzione `void printEmptyMatrix(int n)` che stampa a video una matrice n×n di zeri.
- Un programma, `matr.c`, che stampi a video una matrice 30×30 in cui tutti i valori sono 4. Per farlo, si realizzi una funzione `void printMatrix(int n, int v)` che stampa a video una matrice n×n di `v`.
- Un programma, `morematr.c`, che stampi a video una matrice 30×30 in cui le colonne pari hanno valore 4 e quelle dispari valore 6. Per farlo, si realizzi una funzione `void printMatrixCols(int n, int v1, int v2)` che stampa a video una matrice n×n le cui colonne pari hanno valore v1 e quelle dispari valore v2.
- Un programma, `arrow.c`, che stampi una freccia fatta di 0, come mostrato nell'esempio seguente. Per farlo, si realizzi una funzione `void printArrow(int n)` che stampa a video una freccia con coda lunga `n`. L'esempio sotto è per `n` = 8.

```
          0
0000000000
00000000000
0000000000
          0
```