Università di Ferrara
Dipartimento di Ingegneria

Esercitazioni di
FONDAMENTI DI
INFORMATICA
MODULO B

ESERCITAZIONE 1.0

Tutor

Arnaud Nguembang Fadja: ngmrnd@unife.it

Damiano Azzolini: damiano.azzolini@student.unife.it

Passare argomenti al programma

Quando si invoca un programma è possibile passare degli argomenti ad esso durante la chiamata.

Gli argomenti ricevuti saranno stringhe (array di char) quindi l'insieme di argomenti sarà contenuto in un array di array di char.

Il programma deve poter leggere questi argomenti in modo da poterli gestire.

1. Settare la funzione main in modo tale da considerare gli argomenti passati. Per fare ciò il main deve prendere in ingresso due parametri nel seguente ordine:
 - Un numero intero (n di argomenti + 1) -> `int argc`
 - Un array di stringhe -> `char *argv[]`

Esercizio 1. Passaggio di argomenti

Realizzare un programma C che dato come argomento il nome di un file (di testo), ne visualizzi il contenuto a pagine (20 righe alla volta). Ad ogni visualizzazione chiedere all'utente se vuole visualizzare altre righe o fermarsi. Di seguito una traccia per iniziare:

```
#include<stdio.h>
// ...

main(int argc, char *argv[]) {

    // codice

}
```

Esercizio 1. Testare il programma

Per testare il programma bisogna passare il nome del file come argomento. Abbiamo due possibilità:

1. Eseguire da terminale il programma

A screenshot of a Windows command prompt window titled "Amministratore: C:\Windows\system32\cmd.exe". The window shows a series of commands and their outputs. The user navigates to the directory "C:\Users\Riccardo\Documents\Visual Studio 2010\Projects\tutorial\Debug" and lists the contents. The files listed are "testo.txt", "tutorial.exe", "tutorial.ilc", and "tutorial.pdb". The user then runs "tutorial.exe" without arguments, which results in an error message: "Num. argomenti non corretto". Finally, the user runs "tutorial.exe testo.txt", which outputs a block of Lorem Ipsum text.

```
CA. Amministratore: C:\Windows\system32\cmd.exe
Microsoft Windows [Versione 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Tutti i diritti riservati.

C:\Users\Riccardo>cd "Documents\Visual Studio 2010\Projects\tutorial\Debug"

C:\Users\Riccardo\Documents\Visual Studio 2010\Projects\tutorial\Debug>dir
Il volume nell'unità C è OS
Numero di serie del volume: 0814-E9ED

Directory di C:\Users\Riccardo\Documents\Visual Studio 2010\Projects\tutorial\Debug
05/03/2015  14:57    <DIR>          .
05/03/2015  14:57    <DIR>          ..
05/03/2015  15:00                2.015 testo.txt
05/03/2015  15:08            29.184 tutorial.exe
05/03/2015  15:08           330.368 tutorial.ilc
05/03/2015  15:08           420.864 tutorial.pdb
                4 File                782.431 byte
                2 Directory   155.391.545.344 byte disponibili

C:\Users\Riccardo\Documents\Visual Studio 2010\Projects\tutorial\Debug>tutorial.exe
Num. argomenti non corretto
C:\Users\Riccardo\Documents\Visual Studio 2010\Projects\tutorial\Debug>tutorial.exe tes
to.txt
Loremipsumdolorsitamet,consecteturadipiscielit,sedeiusmodtemporinciduntutlaboreetdolor
magnaaliqua.UtenimContinua (s/n)?s
admininveniam,quisnostrumexercitationemullamcorporissuscipitlaboriosam,nisiutaliquidexe
acommodiconsequatur.QuisauteiureContinua (s/n)?s
reprehenderitinvoluptatevelitessecillumdoloreeufugiatnullapariatur.Excepteursintobcaeca
tcupiditatnonproident,suntin culpaContinua (s/n)?n

C:\Users\Riccardo\Documents\Visual Studio 2010\Projects\tutorial\Debug>
```

Esercizio 1. Testare il programma

Per testare il programma bisogna passare il nome del file come argomento. Abbiamo due possibilità:

2. Settare l'argomento in Visual Studio

Pagine delle proprietà di ConsoleApplication1

Configurazione: Active(Debug) Piattaforma: Active(Win32) Gestione configurazione...

Debugger da avviare: Debugger Windows locale

Comando	\$(TargetPath)
Argomenti del comando	testo.txt
Directory di lavoro	\$(ProjectDir)
Connetti	No
Tipo di debugger	Aut
Ambiente	
Esegui merge ambiente	
Debug SQL	
Tasto di scelta rapida	

Argomenti del comando
Argomenti della riga d

OK Annulla Applica

Click destro sul progetto, selezionare **Proprietà**.
Aprire la tabella **Proprietà di configurazione > Debug**
inserire gli argomenti nella riga **Argomenti del comando**

Esercizio 2. Funzioni come parametri

Si realizzi un programma che calcola l'integrale di una funzione $f : \mathbb{R} \rightarrow \mathbb{R}$ in un intervallo dato $[a,b]$.

Definire una funzione che restituisca il valore dell'integrale, con la seguente interfaccia

```
/*  
 * func funzione integranda  
 * a,b estremi intervallo, N numero sotto-intervalli  
 */  
double rettangoli(double (*func)(double), double a, double  
b, double N);
```

- Il main legge da input a , b , N ; poi chiama `rettangoli` e calcola l'integrale di funzioni reali.
- Definire alcune funzioni da passare a `rettangoli`, per esempio una funzione per calcolare il quadrato e una per calcolare il cubo di un numero reale
- Strutturare il programma su più file

Esercizio 2. Nota

Viene indicato di seguito come specificare il parametro formale di tipo funzione nella dichiarazione della funzione “**rettangoli**”:

`double (*func) (double)`

∨	∨	∨
(1)	(2)	(3)

- (1) = “**tipo** del parametro di ritorno della funzione func”;
- (2) = “**nome** formale della funzione referenziata (puntatore)”;
- (3) = “**lista dei tipi** di parametri di ingresso della funzione func”;

Esercizio 2. Metodo dei rettangoli

Esercizio 3. Variabili globali

Si realizzi un *componente dotato di stato interno* che rappresenta un *contatore*, con operazioni:

- `void reset( void )`
- `void inc( int k )`
- `int getValue( void )`

Il main, che utilizza il contatore, deve in successione:

- resettarlo;
- acquisire da input una sequenza di interi positivi (terminata da 0) e usarli come incremento del contatore.
- stampare lo stato finale
- *La variabile contatore deve essere globale, locale al file contenente le funzioni del contatore.*
- Strutturare il programma su più file

Esercizio 4. Variabili statiche

Si realizzi un *componente dotato di stato interno* che rappresenta un *contatore* (**variabile globale con scope limitato**), con operazioni:

- `void reset( void )`
- `void inc( int k )`
- `int getValue( void )`

Il main, che utilizza il contatore, deve in successione:

- resettarlo;
- acquisire da input una sequenza di interi positivi (terminata da 0) e usarli come incremento del contatore.
- stampare lo stato finale
- *La variabile contatore deve essere **statica**, locale al file contenente le funzioni del contatore.*
- Strutturare il programma su più file

Esercizio 4. Variabili statiche

Una volta concluso si può anche...

- Verificare che lo *scope* della variabile sia solo il file `contatore.c` (“usare” la variabile nel `main` -> errore di compilazione: variabile non definita)