

Strutture Dati, Algoritmi e Complessità


ESERCITAZIONE 1

AA 2023-2024

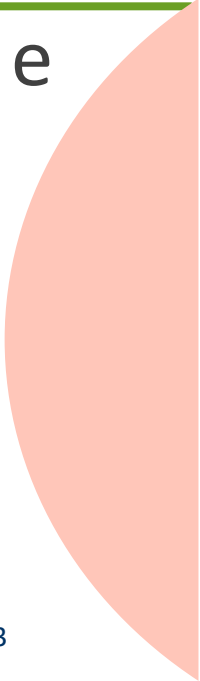
Obiettivi

Gli obiettivi di questa esercitazione sono:

1. prendere confidenza con l'ambiente virtuale BIAR;
2. (ri)visitare semplici comandi della shell Linux/Unix;
3. modificare, compilare ed eseguire codice usando il terminale a riga di comando;
4. usare makefile per gestire il processo di compilazione (C);
5. passaggio di argomenti a riga di comando (Java e C).



Ripasso dei comandi base del terminale Linux e della compilazione di programmi C



Preparare un'area di lavoro al terminale

1. creare cartella/direttorio con `mkdir <nome_scelto>`
2. cambiare direttorio di lavoro con `cd <nome_scelto>`
3. costruire file con `ls -laR / > elenco_files.txt`
 1. ignorare i messaggi di errore
4. verificare contenuto tramite `more elenco_files.txt`
 1. impratichirsi con ' ' e 'q'
 2. testare CTRL +/-
5. creare sottodirettorio con `mkdir test`
6. spostare il file nel sottodirettorio con `mv elenco_files.txt test`
7. creare una copia del file nel direttorio corrente con `cp test/elenco_files.txt .`
 1. attenzione alla differenza fra '\' (usata in Windows) e '/' (usata in linux)

Cercare informazioni

1. cercare gcc nel file con
`grep 'gcc' test/elenco_files.txt`
2. cercare tutti i file che contengono la parola gcc con
`find / -name "gcc" -print`
(ignorare gli errori)
3. provare autonomamente varianti delle due ricerche

vi-vim

potente editor

- `vi pippo.txt` crea il file (vuoto) `pippo.txt`

tre modalità:

- inserimento
- comandi (modalità di default)
- comandi estesi
- per uscire da una modalità premere ESC (eventualmente più volte) per andare nella modalità comandi

'a' e 'i' entrano in modalità inserimento o append, 'X' cancella carattere sottostante

- molte varianti fra cui 'A' e 'I'

':' entra in modalità comandi estesi (dalla modalità comandi)

la dotazione di features è enorme, copia/incolla/taglia, macro ecc.

il '.' (in modalità comandi) ripete l'ultimo comando eseguito

":q!" per uscire senza salvare

Compilare e capire argc e argv

Usando un text editor qualsiasi (anche `vi/vim`) inserire il seguente codice C creando il file `args.c`

```
#include <stdio.h>
int main(int argc, char *argv[]) {
    printf("argc vale %d\n", argc);
    for(int i = 0; i < argc; i++)
        printf("argv[%d] = %s\n", i, argv[i]);
    return 0;
}
```

1. compilare con `gcc args.c` ed eseguire con `./a.out 1 2 3 4`
2. compilare con `gcc args.c -o pippo` ed eseguire con `./pippo a b c`
3. abituarsi a usare sempre il flag `-g` (utile per il debugger, che sarà visto in altro corso)

makefile (più avanzato)

```
CC = gcc
```

```
CFLAGS = -Wall -g
```

```
SRCS = args.c
```

```
MAIN = main
```

```
RM = rm
```

```
all: $(MAIN); chmod u+x $(MAIN)
```

```
$(MAIN):
```

```
$(CC) $(CFLAGS) -o $(MAIN) $(SRCS)
```

```
clean:
```

```
$(RM) *.o $(MAIN)
```

nome del file: makefile

<variabile1> = <espressione1>

<variabile2> = <espressione2>

<variabile3> = <espressione3>

<target>: <prerequisiti>

<tab><comando1>


<tab><comando2>

... *

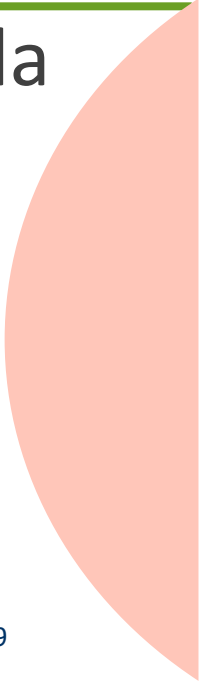
multi target possibili

basta scrivere make

o make <target>



Compilare ed eseguire programmi in Java e C da riga di comando



Esercizio 1

1. Scaricare il codice che implementa l'algoritmo di Bubble Sort (disponibile sia in C che in Java)
2. Per la compilazione: usare `gcc` / `javac`. In C usare `makefile`
3. Per l'esecuzione: invocazione a riga di comando (C) oppure `java <class file>`

NB. Nella versione di base di Bubble Sort fornita, l'array da ordinare è definito staticamente all'interno del codice.

Esercizio 2

Estendere il codice in modo da leggere l'array da ordinare direttamente da riga di comando (`argc/argv` in C o `args[]` in Java)

Esercizio 3

Estendere il codice in modo da leggere l'array da ordinare generando un array di interi casuali di dimensione data.

- gli argomenti passati al programma saranno la stringa "rnd" seguita da spazio e da un intero che specifica la dimensione dell'array.
- In C la generazione di un array casuale di dimensione data è già realizzata dalla funzione `randArray`,
- In Java occorre usare la classe `java.util.Random`, dopo averne controllato la documentazione

Esercizio 4

Si aggiunga poi del codice che consenta di misurare il tempo di esecuzione del BubbleSort.

- Tale funzionalità è già implementata nel codice C
- mentre in Java si può implementare usando ad esempio `System.currentTimeMillis` (verificare la documentazione)