Student Name:
Student ID:

LabReSiD25
Hands-On 9

Università degli Studi di Messina

Introduzione

Il presente è HandsOn è pensato per familiarizzare, **in vista dei progetti di fine corso**, con la programmazione modulare (cioè divisa su più file) e con l'utilizzo del programma `make` per l'automazione della compilazione. E' richiesta la scrittura di un report con le osservazioni richieste, da inviare al docente.

1 Analisi del codice: Tipi di variabili e tabella dei simboli

Compilare, senza linkare, il file sottostante ed osservare la tabella dei simboli con `nm` comprendendo e giustificando ogni colonna dell'output `(indirizzo, tipologia (b,d,t,u,r), nome del simbolo), man nm`.

```c
#include <stdio.h>

int a;
static int b = 10;

void funzione() {
    static int c = 5;
    int d = 20;
    c++;
    d++;
}

int main() {
    funzione();
    return 0;
}
```

2 Analisi del codice: Compilazione modulare

Compilare con `make` il programma modulare sottostante ed osservare la tabella dei simboli generate con `nm` tramite `make check` comprendendo e giustificando ogni colonna dell'output `(indirizzo, tipologia (b,d,t,u,r), nome del simbolo), man nm`. Aggiungere/eliminare casualmente delle variabili ed usare `size --format=GNU` per osservare la crescita dei segmenti `text,data,bss`.

header.h

```
#ifndef HEADER_H
#define HEADER_H

extern int var_extern;

void modifica_var_extern(void);
void stampa_var_extern(void);

#endif
```

file1.c

```
#include <stdio.h>
#include "header.h"

int var_extern = 10;
static int var_static = 100;

void modifica_var_extern(void) {
    var_extern++;
    var_static += 10;
    printf("file1: var_extern=%d, var_static=%d\n", var_extern, var_static);
}
```

file2.c

```
#include <stdio.h>
#include "header.h"

void stampa_var_extern(void) {
    printf("file2: var_extern=%d\n", var_extern);
}
```

main.c

```
#include "header.h"

int main() {
    modifica_var_extern();
    stampa_var_extern();

    modifica_var_extern();
    stampa_var_extern();

    return 0;
}
```

Makefile

```
  CC = gcc
CFLAGS = -g -Wall
TARGET = programma
OBJS = main.o file1.o file2.o

all: $(TARGET)

$(TARGET): $(OBJS)
  $(CC) $(CFLAGS) -o $@ $^

%.o: %.c
  $(CC) $(CFLAGS) -c $<

check:
  @echo "\n=== Simboli file1.o ==="
  nm file1.o
  @echo "\n=== Simboli file2.o ==="
  nm file2.o
  @echo "\n=== Simboli main.o ==="
  nm main.o

clean:
  rm -f $(OBJS) $(TARGET)

.PHONY: all clean check
```