Sistemi di Calcolo (A.A. 2017-2018)
Corso di Laurea in Ingegneria Informatica e Automatica
Sapienza Università di Roma

D

Compito di esonero (14/11/2017) – Durata 1h 30'

Inserire nome, cognome e matricola nel file `studente.txt`.

Parte 1 (programmazione IA32)

Nella directory `es1D`, si traduca in assembly IA32 la seguente funzione C scrivendo un modulo `es1D.s`:

```
void fetch(short u, short* p);

void update(short* v, int n) {
    int i;
    short x;
    for (i=0; i<n; ++i) {
        fetch(v[i], &x);
        v[i] = x;
    }
}
```

L'unico criterio di valutazione è la correttezza, cioè l'equivalenza semantica tra il programma tradotto e quello C di partenza. Generare un file eseguibile `es1D` con `gcc -m32 -g`. Per i test, compilare il programma insieme al programma di prova `es1D-main.c` e al modulo `fetch.s`.

Nota: non modificare in alcun modo `es1D-main.c` e `fetch.s`.

Parte 2 (programmazione IA32)

Nella directory `es2D`, si traduca in assembly IA32 la seguente funzione C scrivendo un modulo `es2D.s`:

```
int veclen(int x, int y, int* d) {
    return x >= 0 && y >= 0 && x*x+y*y < (*d)*(*d);
}
```

L'unico criterio di valutazione è la correttezza, cioè l'equivalenza semantica tra il programma tradotto e quello C di partenza. Generare un file eseguibile `es2D` con `gcc -m32 -g`. Per i test, compilare il programma insieme al programma di prova `es2D-main.c`.

Nota: non modificare in alcun modo `es2D-main.c`.

Parte 3 (ottimizzazione del work)

Nella directory `es3D`, si crei una versione ottimizzata del seguente modulo `es3D.c` scrivendo un modulo `es3D-opt.c`:

```
#include "es3D.h"
#include "list.h"
#include <stdlib.h>

struct charset {
    list* l;
};
```

```
charset* charset_new() {
    charset *s = malloc(sizeof(charset));
    s->l = list_new();
    return s;
}

void charset_add(charset* s, unsigned char c) {
    if (!list_contains(s->l, c)) list_add(s->l, c);
}

int charset_contains(charset* s, unsigned char c) {
    return list_contains(s->l, c);
}

void charset_delete(charset* s) {
    list_delete(s->l);
    free(s);
}
```

Il modulo implementa una struttura dati per rappresentare insiemi di caratteri con le usuali operazioni di inserimento e verifica di appartenenza all'insieme. *Suggerimento*: si osservi che i caratteri sono valori numerici in un piccolo intervallo. **Verificare che la soluzione ottimizzata stampi gli stessi risultati di quella originaria!**

Per compilare, usare **sempre** le opzioni `-m32 -O1` e il programma di prova `es3D-main.c` insieme al modulo `list.c`.

Ai fini dell'ottimizzazione:

1. usare `gprof` per identificare le porzioni più onerose computazionalmente
2. esaminare il modulo `es3D.s` generato a partire da `es3D.c` con `gcc -S -O1` (e già fornito) per capire quali ottimizzazioni siano già state effettuate dal compilatore

Alla fine del compito, la directory **dovrà contenere** i seguenti file non presenti in origine:

1. `es3D`, eseguibile ottenuto da `es3D.c`
2. `es3D-pg`, eseguibile ottenuto da `es3D.c` con l'opzione `-pg`
3. `gmon.out`, report binario generato da `gprof`
4. `es3D-pg.txt`, report testuale generato da `gprof`
5. `es3D-opt`, eseguibile ottenuto da `es3D-opt.c`

Rispondere alle seguenti domande nel file `es3D-risposte.txt`:

1. descrivere le ottimizzazioni applicate e dire perché si ritiene che siano efficaci
2. riportare i tempi di esecuzione real di `es3D` ed `es3D-opt` misurati con il comando `time` e mediati su tre esecuzioni distinte (trial)
3. riportare lo speedup ottenuto (rapporto dei tempi medi calcolati al punto 2)

Parte 4 (quiz)

Si risponda ai seguenti quiz, inserendo le risposte (A, B, C, D o E per ogni domanda) nel file `es4A.txt`. **Una sola risposta è quella giusta**. Rispondere E equivale a non rispondere (0 punti).

Domanda 1 (endianness)

Si assuma di operare in un'architettura IA32 sul seguente frammento di memoria:

Indirizzo	0x1000	0x1001	0x1002	0x1003
Contenuto	0x11	0x22	0x33	0x44

Eseguendo le seguenti istruzioni:

```
movl $0xAABBCCDD, %eax
movw %ax, 0x1002            # scrive all'indirizzo 0x1002
movw 0x1001, %ax            # legge dall'indirizzo 0x1001
```

Cosa conterrà il registro `%ax`?

A	0x33DD	**B**	0xDD22
C	0xCC33	**D**	0xCC22

Motivare la risposta nel file `M1.txt`. **Risposte non motivate saranno considerate nulle.**

Domanda 2 (estensione bit)

Si assuma di eseguire in un'architettura IA32 il seguente frammento di codice:

```
movl $0xAABBCCDD, %eax
movw $0x5FFF, %cx
movzbw %cl, %ax
```

Cosa conterrà il registro `%eax`?

A	0xFFFFFFFF	**B**	0xAABB00FF
C	0x000000FF	**D**	0xAABBFFFF

Motivare la risposta nel file `M2.txt`. **Risposte non motivate saranno considerate nulle.**

Domanda 3 (uso del compilatore)

Quale dei seguenti comandi permette di ottenere il codice assembly generato dal compilatore **`gcc`** per un modulo C contenuto in **`f.c`**?

A	`gcc -g -S f.c`	**B**	`gcc -S -o f.c`
C	`gcc -o -S f.c`	**D**	`gcc -OS f.c`

Motivare la risposta nel file `M3.txt`. **Risposte non motivate saranno considerate nulle.**

Domanda 4 (ottimizzazione)

Si consideri il seguente programma C e il corrispettivo codice assembly emesso dal compilatore (livello ottimizzazione 2):

```
int f(unsigned x) {
    int k, s = 0;
    for (k = 0; k < x; k++) {
        s += 5*x + k;
    }
    return s;
}
```

```
ff:
    pushl   %ebx
    movl    8(%esp), %ecx
    testl   %ecx, %ecx
    je  .L4
    leal    (%ecx,%ecx,4), %ebx
    xorl    %eax, %eax
    xorl    %edx, %edx
.L3:
    addl    %ebx, %eax
    addl    %edx, %eax
    addl    $1, %edx
    cmpl    %ecx, %edx
    jne .L3
    popl    %ebx
    ret
.L4:
```

	`xorl %eax, %eax`
	`popl %ebx`
	`ret`

Quale tra le seguenti ottimizzazioni è stata effettuata dal compilatore?

A	augmentation	**B**	loop inviariant code motion
C	common subexpression elimination	**D**	cortocircuitazione

Motivare la risposta nel file `M4.txt`. **Risposte non motivate saranno considerate nulle.**