به نام خدا

دانشکده مهندسی کامپیوتر

درس ساختار و زبان کامپیوتر
نیم‌سال اول ۰۴-۰۳
استاد: دکتر اسدی

تمرین سری ششم

- پرسش‌های خود را در سامانه CW و تالار مربوط به تمرین مطرح نمایید.
- پاسخ سوالات را تایپ نمایید.
- اسکرین‌شات‌ها، عکس‌ها، فایل‌های مربوط به سوال عملی، گزارش تمرینات عملی و PDF قسمت تئوری را در پوشه با نام به فرمت HWNUM_StudentID1_StudentID2 ذخیره نمایید. سپس آن را zip نمایید و در صفحه درس بارگذاری نمایید. به عنوان مثال یک فایل بارگذاری شده قابل قبول باید دارای فرمت HW1_400123456_403123456.zip باشد.
- هر دانشجو می‌تواند حداکثر دو تمرین را با دو روز تأخیر بدون کاهش نمره ارسال نماید.
- تمرینات عملی به صورت گروه‌های دو نفر تحویل داده شود.
- هر دو عضو گروه موظف هستند تمرینات خود را بارگذاری کنند.
- عواقب عدم تطابق بین پاسخ دو عضو گروه برعهده خودشان است.
- تحویل تمرین به صورت انگلیسی مجاز نیست. در صورت تحویل تمرین به صورت انگلیسی (حتی بخشی از تمرین) نمره تمرین موردنظر صفر در نظر گرفته می‌شود.
- در صورت مشاهده تقلب برای بار اول نمره هر دو طرف صفر می‌شود. در صورت تکرار نمره کل تمرینات صفر خواهد شد.
- استفاده از ابزارهایی مانند ChatGPT به منظور ابزار کمک آموزشی مجاز است به شرط آن که به خروجی آن اکتفا نشود.
- توجه شود که پروژه نهایی درس در گروه‌های چهار نفر تحویل گرفته می‌شود.
- سوالات با عنوان **اختیاری** نمره‌ای ندارند اما جواب دادن به آن‌ها کمک به‌سزایی در یادگیری درس می‌کند.

تمارین تئوری

۱. کد زیر که به زبان اسمبلی MIPS نوشته‌شده است را بخوانید. ابتدا بگویید کد چکار می‌کند و معادل آن را به زبان C بنویسید.

حالا دقت کنید که در این کد تعدادی ایراد منطقی و همچنین تخطی از روند استاندارد فراخوانی توابع در MIPS [۱] دیده می‌شود. این موارد را مشخص کنید و بگویید چگونه باید اصلاح شوند. نیازی به نوشتن دوباره‌ی کد نیست، صرفا توضیح دهید چگونه می‌توان مشکل را برطرف کرد.

```
.data
newline: .asciiz "\n"

.text
.globl main

main:
    li $a0, 135
    li $a1, 1
    jal weird_func
    move $a0, $v0
    li $v0, 1
    syscall
    la $a0, newline
    li $v0, 4
    syscall
    li $v0, 10
    syscall

weird_func:
    beq $a0, 0, base_case
    addi $sp, $sp, -12
    sw $ra, 0($sp)
    sw $a1, 4($sp)
    sw $a0, 8($sp)
    li $t0, 10
    div $a0, $t0
    mfhi $t1
    mflo $t2
    xori $a1, $a1, 1
    move $a0, $t2
    jal weird_func
    lw $a1, 4($sp)
    lw $a0, 8($sp)
    lw $ra, 0($sp)
    bne $a1, $zero, add_case
    sub $v0, $v0, $t1
    j exit_case
add_case:
    add $v0, $v0, $t1
    j exit_case
base_case:
    li $v0, 0
    addi $sp, $sp, 12
exit_case:
```

[۱] MIPS Calling Convention

```
    addi $sp, $sp, 12
    jr $ra
```

۲. در کد زیر بعد از اجرای هر دستور، مقداری که در ثبات a0 ریخته می‌شود را مشخص کنید. توضیح دهید که به چه علت این مقادیر داخل ثبات ذخیره می‌شود.
حال فرض کنید که num_2 به صورت Little Endian ذخیره شده است. در این حالت مقدار ثبات در کدام دستورات نسبت به حالت اول تغییر می‌کند؟ توضیح دهید.

```
.data
num_1:  .asciiz "1234"
num_2:  .word   1234
.text
main:
    lw  $a0, num_1
    lb  $a0, num_1
    lb  $a0, num_1+1
    lb  $a0, num_1+2
    lb  $a0, num_1+3
    lh  $a0, num_1
    lh  $a0, num_1+1
    lh  $a0, num_1+2
    lw  $a0, num_2
    lb  $a0, num_2
    lb  $a0, num_2+1
    lb  $a0, num_2+2
    lb  $a0, num_2+3
    lh  $a0, num_2
    lh  $a0, num_2+1
    lh  $a0, num_2+2
```

۳. در این سوال به شما یک کد به زبان C داده شده است. شما باید این کد را به زبان MIPS پیاده سازی کنید و سپس ماشین کد آن را با فرض اینکه شروع این کد از آدرس 0x00000000 است. فرض کنید که بخش text و data به صورت پیوسته در حافظه قرار می‌گیرند. همچنین توجه کنید که دقیقا همین کد باید پیاده سازی شود.

```
#include <stdio.h>

int func2(int n, int m) {
    return 2 * m * (n/2);
}

int func1(int n, int m) {
    if (n == 0 || m == 0)
        return 1;
    return n + func1(n--, --m) + func2(n, m--);
}

int main() {
    int n = 5;
    int m = 6;
    func1(n, m);
    return 0;
}
```

۴. شما با کد زیر در بخش داده یک برنامه MIPS مواجه شده‌اید. فرض کنید اولین داده در آدرس 0x1000 قرار دارد. به سوالات زیر پاسخ دهید:

```
    .data
    .align 2
data:
    .word  0x12345678
    .half  0x9ABC
    .byte  0xDE
    .byte  0xF0
```

(آ) چرا وجود رهنمود[۲] align 2. در بخش داده برنامه لازم است؟

(ب) آدرس ذخیره‌سازی هر داده در حافظه را مشخص کنید.

(ج) فرض کنید از دستور lw برای دسترسی به کلمه ذخیره شده در آدرس 0x1002 اقدام می‌کنیم. در صورت موفقیت چه مقداری در ثبات مقصد ذخیره می‌شود؟ در صورت عدم موفقیت چه اتفاقی رخ می‌دهد و چرا؟

(د) کدی به زبان اسمبلی MIPS بنویسید که دستور lw از یک آدرس misaligned را شبیه‌سازی کند. آیا Endianness تفاوتی در کد ایجاد می‌کند؟ توضیح دهید.

(ه) مقدار ذخیره‌شده در ثبات مقصد را پس از اجرای هر یک از دستورات زیر مشخص کنید.

```
    la      $t0, data
    lh      $t1, 4($t0)
    lb      $t2, 6($t0)
    lbu     $t3, 7($t0)
    lhu     $t4, 4($t0)
```

۵. با توجه به کد زیر به سؤالات مطرح شده پاسخ دهید.

```
0| fun:
1| addi sp, sp, -16
2| sw ra, 12(sp)
3| sw a0, 8(sp)
4| li t0, 2
5| blt a0, t0, base_case
6| addi a0, a0, -1
7| jal fun
8| sw v0, 4(sp)
9| lw a0, 8(sp)
10| addi a0, a0, -2
11| jal fun
12| lw t0, 4(sp)
13| add v0, v0, t0
14| sw v0, 4(sp)
15| lw a0, 8(sp)
16| addi a0, a0, -3
17| jal fun
18| lw t0, 4(sp)
19| add v0, v0, t0
20| addi v0, v0, 2
21| j end_fun
22| base_case:
23| li v0, 1
```

[۲] directive

```
24| end_fun:
25| lw ra, 12(sp)
26| addi sp, sp, 16
27| jr ra
```

۱. کد اسمبلی داده شده را به زبان c ترجمه کنید.
۲. تابع fun(۱۰) را اجرا کنید و مقدار آنرا خروجی دهید.
۳. در قسمت قبل چند بار تابع fun صدا زده شده است؟
۴. برای بهینه سازی اجرای این تابع چه کاری را پیشنهاد می‌کنید؟

.

۶. دو تابع زیر را به زبان اسمبلی میپس ترجمه کنید. پیشنهاد می‌شود از سایت godbolt.org کمک بگیرید. (در سایت، از کامپایلر (codespace) mips gcc 9.3.0 استفاده کنید.)

```
int f1(int a, int b, int c, int d, int e, int f) {
    return a * b * c * d * e * f;
}
```

```
int f2(int *a) {
    int ans = 1;
    for (int i = 0; i < 6; i++) {
        ans *= a[i];
    }
    return ans;
}
```

بهینه‌سازی‌هایی را که کامپایلر در ترجمه این دو تابع به اسمبلی انجام داده‌است، بررسی کنید (از پرچم‌های مراتب مختلف بهینه‌سازی در کامپایل استفاده کنید).

تمارین عملی

۱. برنامه‌ای به زبان اسمبلی MIPS بنویسید که بررسی کند که یک زیررشته چندبار در هر پیشوند یک رشته تکرار شده است. در ورودی به ترتیب طول زیررشته، زیررشته، طول رشته و رشته به شما داده می‌شود. در خروجی به ترتیب در هر خط تعداد تکرار آن زیررشته در پیشوندهای به طول ۱، ۲ و ... رشته را نشان دهید. تضمین می‌شود که طول رشته‌ها در بازه اعداد ۳۲ بیتی می‌باشد. (برای استفاده بهینه از حافظه، پیشنهاد می‌شود از حافظه پویا استفاده شود)
نمونه ورودی:

```
2
ab
6
aabbab
```

خروجی:

```
0
0
1
1
1
2
```

توضیح: در خط اول خروجی تعداد تکرار زیررشته در پیشوند به طول یک رشته چاپ شده است. در خط دوم تعداد تکرار در پیشوند به طول دو، در خط سوم تعداد تکرار در پیشوند به طول ۳ نوشته شده است که برابر با یک می‌باشد (aab که یک ab در خود دارد) و به همین ترتیب

۲. برنامه‌ای بازگشتی به زبان اسمبلی MIPS بنویسید که با ورودی گرفتن عدد n، تمامی اعداد دودویی n رقمی که هیچ دوتا رقم ۱ متوالی ندارند را چاپ کند. ترتیب چاپ کردن این اعداد باید به طور نزولی باشد.
نمونه ورودی:

```
4
```

نمونه خروجی:

```
1010
1001
1000MIPS
0101
0100
0010
0001
0000
```

۳. تابع TAK برای آزمون قدرت پردازشی و تحلیل الگوریتم‌های بازگشتی پیچیده استفاده می‌شود. معمولاً در آزمون‌های عملکرد سیستم‌ها و شبیه‌سازی مسائل محاسباتی سنگین به کار می‌رود.

برنامه‌ای به زبان اسمبلی MIPS بنویسید که سه عدد در سه خط از ورودی بخواند و تابع بازگشتی TAK را روی آنها اجرا کند و نتیجه را چاپ کند.

```
int tak(int x, int y, int z) {
    if (x <= 0) {
        return y;
    } else if (y <= 0) {
        return z;
```

```
    } else if (z <= 0) {
        return tak(x - 1, y - 1, z);
    } else {
        return tak(x - 1, tak(y - 1, z, x), tak(y, z - 1, x));
    }
}
```

۴. یک برنامه به زبان MIPS بنویسید که یک ماتریس ۴x۴ را که به‌صورت یک آرایه ۱۶ بایتی در حافظه ذخیره شده است، به اندازه ۹۰ درجه در جهت عقربه‌های ساعت بچرخاند. هر عنصر ماتریس ۱ بایت است.

۵. برنامه‌ای بنویسید که یک عدد حداکثر ۱۰ رقمی را از کاربر ورودی گرفته و به صورت بازگشتی پالیندروم بودن یا نبودن آن را بررسی کند.(در صورت بازگشتی نبودن جواب شما، نصف نمره این مسئله را از دست خواهید داد.)