تمرین فصل دوم - دستورات: زبان کامپیوتر

زهره سورانی
مهدی حق‌وردی

چکیده

سوالات فصل اول کتاب، که آموزش ISAی MIPS است، برای شما تالیف شده‌اند.
پاسخ هر سوال را در قسمت مربوط آنها در کوئرا به صورت PDF به صورت تایپ شده، یا دست‌نویس خوش‌خط و خوانا آپلود کنید.
هر سوال دارای یک پاورقی‌ست که طراح آن سوال را مشخص می‌کند، برای پرسیدن سوالات خود به طراح هر سوال مراجعه کنید.
پس از پایان یافتن زمان ارسال تمرین، پاسخ‌های این تمرین در آدرس زیر قرار خواهد گرفت.

`https://github.com/mahdihaghverdi/arch-questions-answers/blob/main/instructions-language-of-the-computer`

فهرست مطالب

۱ C به Assembly

تابع زیر را به زبان اسمبلی بنویسید.

```
void copy(int a[], int b[], int n){
    int i;
    for(i=0; i!=n; i++) {
        a[i]=b[i];
    }
}
```

۲ عملیات‌های منطقی

سوالات زیر را بر اساس جدول زیر که محتویات رجیسترهای $t0 و $t1 را داراست، بنویسید.

$t0 = 0xAAAAAAAA	$t1 = 0x12345678

۱. مقدار $t2 بعد از اجرای کد زیر چقدر است؟ پاسخ را به صورت hexadeciaml بنویسید.

```
sll  $t2, $t0, 44
or   $t2, $t2, $t1
```

۲. مقدار $t2 بعد از اجرای کد زیر چقدر است؟ پاسخ را به صورت hexadeciaml بنویسید.

```
sll  $t2, $t0, 4
andi $t2, $t2, -1
```

۳. مقدار $t2 بعد از اجرای کد زیر چقدر است؟ پاسخ را به صورت hexadeciaml بنویسید.

```
srl  $t2, $t0, 3
andi $t2, $t2, 0xFFEF
```

۳ عملیات‌های شرطی Branches

۱.۳ مقدار نهایی چیست؟

کدهایی زیر را در نظر بگیرید

1.
```
LOOP:  addi  $s2, $s2, 2
       subi  $t1, $t1, 1
       bne   $t1, $0, LOOP
DONE:
```

2.
```
LOOP:  slt   $t2, $0, $t1
       beq   $t2, $0, DONE
       subi  $t1, $t1, 1
       addi  $s2, $s2, 2
       j     LOOP
DONE:
```

فرض کنید که مقدار اولیه‌ی رجیستر $t1 برابر است با 10 و مقدار اولیه رجیستر $s2 برابر با صفر، مقدار نهایی $s2 چند است؟

۲.۳ کد اسمبلی را بنویسید.

کد اسمبلی کد زیر را بنویسید.

```
for(i=0; i<a; i++)
   for(j=0; j<b; j++)
      D[4*j] = i + j;
```

۴ توابع

۱.۴ تابع positive

کد اسمبلی تابع زیر را بنویسید.

```
int positive(int a, int b) {
   if (addit(a, b) > 0)
     return 1;
   else
      return 0;
}
int addit(int a, int b) {return a+b;}
```

۲.۴ In-line کردن تابع

کامپایلرهای زبان‌های برنامه‌نویسی می‌توانند توابعی را in-line کنند. این به این معناست که کد تابع را در دل جایی که نوشته شده‌اند، تزریق میکنند و از یک function call جلوگیری می‌کنند. حال شما کد تابع addit را in-line کنید.

۳.۴ چندین بار صدا زده شدن تابع

کد زیر را به اسمبلی بنویسید. (نیازی به دانستن بدنه‌ی تابع func نیست، صرفا آن را به صورت یک label تعریف کنید)

```
int f(int a, int b, int c, int d){
    return func(func(a,b), c+d);
}
```

۵ توابع بازگشتی

۱.۵ تابع فاکتوریل

کد اسمبلی زیر، یک کد دارای اشتباه از پیاده‌سازی فاکتوریل است، آنرا اصلاح کنید. مقدار اولیه پاس داده شده به تابع، در رجیستر $a0 ذخیره شده و نتیجه در رجیستر $v0 ذخیره می‌شود.

```
FACT:   sw      $ra, 4($sp)
        sw      $a0, 0($sp)
        addi    $sp, $sp, -8
        slti    $t0, $a0, 1
        beq     $t0, $0, L1
        addi    $v0, $0, 1
        addi    $sp, $sp, 8
        jr      $ra

L1:     addi    $a0, $a0, -1
        jal     FACT
        addi    $sp, $sp, 8
        lw      $a0, 0($sp)
        lw      $ra, 4($sp)
        mul     $v0, $a0, $v0
        jr      $ra
```

۲.۵ تابع فیبوناچی

کد اسمبلی زیر، یک کد دارای اشتباه از پیاده‌سازی دنباله‌ی فیبوناچی است، آنرا اصلاح کنید. مقدار اولیه پاس داده شده به تابع، در رجیستر $a0 ذخیره شده و نتیجه در رجیستر $v0 ذخیره می‌شود.

```
1  FIB:   addi   $sp, $sp, -12
2         sw     $ra, 0($sp)
3         sw     $s1, 4($sp)
4         sw     $a0, 8($sp)
5         slti   $t0, $a0, 1
6         beq    $t0, $0, L1
7         addi   $v0, $a0, $0
8         j      EXIT
9
10 L1:    addi   $a0, $a0, -1
11        jal    FIB
12        addi   $s1, $v0, $0
13        addi   $a0, $a0, -1
14        jal    FIB
15        add    $v0, $v0, $s1
16
17 EXIT:  lw     $ra, 0($sp)
18        lw     $a0, 8($sp)
19        lw     $s1, 4($sp)
20        addi   $sp, $sp, 12
21        jr     $ra
```