

2023/9/30 26 答 答 答

1. 下列那一条指令是正确的: C。

A movb \$0xE, (%ebx) B movl %rax, (%rsp)

C movq %rax, %rdx D movw (%rax), 4(%rsp)

2. 1) mov (%eax, %eax, 4), %eax

2) lea (%eax, %eax, 4), %eax

上面指令中的那一条会产生如下结果: %eax = 5 * %eax?(C)

A 1)和 2)都不会 B 1) C 2) D 1)和 2)都会

3. 下列指令中不会改变 PC 寄存器内容的是 A。

A ADD B JMP C CALL D RET

4. 已知 int P[M][N]和 int Q[N][M], 有以下函数:

```
int addfun( int i,int j){  
    return P[i][j]+Q[j][i];  
}
```

对应汇编代码如下, 请问 M 和 N 分别是多少?

addfun:

```
movl    %edi, %edx  
shl     $2, %edx  
addl    %esi, %edx  
movl    %esi, %eax  
shll    $2, %eax  
addl    %eax, %edi  
movl    Q(,%rdi,4), %eax  
addl    P(,%rdx,4), %eax  
ret
```

M=4, N=4

~~4, 4~~

5. 已知函数 Sum_fun 的 C 语言代码及其对应的 x86-64 汇编代码框架, 请补

齐缺失的汇编代码

```
void Sum_fun (int *p)  
{  
    int a[4] = {1, 2, 4, 8}, i = 0;  
    do {  
        *p += a[i];  
        i = i+1;  
    }  
}
```



```

    } while(i < 4)
}

    subq    $0x10, %rsp
    movl    $0x08, 12(%rsp)
    movl    $4, 8(%rsp)           # a[2] = 4
    movl    $2, 4(%rsp)
    movl    $1, (%rsp)
    movl    $0, %eax
    movl    $0, %ecx
.L1
    movslq  %eax, %rax
    addl    (%rsp,%rax,4),%ecx    # sum += a[i]
    addq    $1, %rax              # i++
    cmpq    $4, %rax              # compare i with 4
    jl      .L1
    addl    %ecx, (%rdi)
    addq    $0x10, %rsp           # Deallocate stack
    ret

```

6, 考虑如下一个C函数和对应的x86-64汇编代码。C代码若有缺失, 请在下划线处补全, 并且填写下面的跳转表。

```

int s_f(int a, int b)
{
    switch(a)
    {
        case 210:
            b *= 13;
            break
        case 213:
            b = 18243;

            _____
        case 214:
            b *= b;
            break
        case 216:

```

```

40045c < s_f >:
40045c: lea     -0xd2(%rdi), %eax
400462: cmp     $0x9, %eax
400465: ja      40048a < s_f +0x2e>
400467: mov     %eax, %eax
400469: jmpq    *0x400590(, %rax, 8)
400470: lea     (%rsi,%rsi,2),%eax
400473: lea     (%rsi,%rax,4),%eax
400476: retq
400477: mov     $0x4743, %esi
40047c: mov     %esi, %eax
40047e: imul    %esi, %eax
400481: retq
400482: mov     %esi, %eax
400484: sub     %edi, %eax
400486: retq
400487: add     $0xd, %esi
40048a: lea     -0x9(%rsi), %eax
40048d: retq

```



```

        case 218:
            b -= a;
            break
        case 219:
            b += 13;

        default:
            b -= 9;
    }
    return b;
}

```

提示:

0xd2 = 210
0x4743 = 18243

跳转表:

0x400590: <u>0x400470</u>	0x400598: <u>0x40048a</u>	_____
0x4005a0: <u>0x40048a</u>	0x4005a8: <u>0x400477</u>	_____
0x4005b0: <u>0x40047c</u>	0x4005b8: <u>0x40048a</u>	_____
0x4005c0: <u>0x400482</u>	0x4005c8: <u>0x40048a</u>	_____
0x4005d0: <u>0x400482</u>	0x4005d8: <u>0x400487</u>	_____

7. 函数 P 与结构体 test 的定义如下, 并完成以下的问题:

```

struct test {
    char *a;
    char b;
    int c;
};

long P(long x, long y, struct test *s)
{
    long u = Q(y);
    long v = Q(x);
    return u+v+(s->c);
}

1  P:
2      pushq %rbp
3      pushq %rbx
4      pushq %r12
5      subq $8, %rsp
6      movq %rdx, %r12
7      movq %rdi, %rbp

```



```

8    movq  %rsi, %rdi
9    call  Q
10   movq  %rax, %rbx
11   movq  %rbp, %rdi
12   call  Q
13   addq  %rbx, %rax
14   addq  16(%r12), %rax
15   addq  $8, %rsp
16   popq  %r12
17   popq  %rbx
18   popq  %rbp
19   ret

```

补全 P 函数对应的汇编代码中的缺失部分；

