

I. Procedure

1. Calling Convention:

Assume we have a function declared as:

```
int f(int a, short b, char c, int d,  
      long e, int f, char g, int h);
```

And we call f in g:

```
int g(int a, int b) {  
    int ret = f(0, 1, 2, a, -31, 4, 5, b);  
    return ret + 3;  
}
```

Please fill the blanks in the assembly code for g:

(Note: all registers used to pass arguments are caller-saved)

```
pushq   %rbp  
movq    %rsp, %rbp  
pushq   %rsi  
pushq   $5  
movl    $4, %r9d (movq $4, %r9)  
movq    $-3, %r8  
movl    %edi, %ecx (movq %rdi, %rcx)  
movb    $2, %dl (movw .../movl .../movq ...)  
movw    $1, %si (movl .../movq ...)  
movl    $0, %rdi  
call    f  
addq    $16, %rsp  
addl    $3, %eax  
leave  
ret
```

2. Stack Frame

Assume we have a function f:

```
int f() {  
    long h = 1;  
    char i = 2;  
    int j = 3;  
    short k = 4;  
    return 0;  
}
```

And the assembly code is shown below:

```
push    %rbp
mov     %rsp, %rbp
subq    _____, %rsp
movq    $1, _____(%rbp)
movb    $2, _____(%rbp)
movl    $3, _____(%rbp)
movw    $4, _____(%rbp)
movl    $0, _____
leave
ret
```

1) Please complete the following assembly code:

```
movq    8(%rbp), %rax    // Load return address into %rax
movq    0(%rbp), %rax    // Load saved rbp into %rax
```

2) If the compiler allocates local variables one by one (&h > &i > &j > &k), please fill the blanks in the assembly code of f.

(Not the only solution)

```
push    %rbp
mov     %rsp, %rbp
subq    $24, %rsp
movq    $1, $-8(%rbp)
movb    $2, $-9(%rbp)
movl    $3, $-16(%rbp)
movw    $4, $-18(%rbp)
movl    $0, %eax
leave
ret
```

3) If the compiler optimized the stack usage (reorder local variables to consume minimum stack memory), please fill the blanks in the assembly code of f.

(Not the only solution)

```
push    %rbp
mov     %rsp, %rbp
subq    $16, %rsp
movq    $1, $-8(%rbp)
movb    $2, $-9(%rbp)
movl    $3, $-16(%rbp)
movw    $4, $-12(%rbp)
movl    $0, %eax
leave
ret
```

(Note: %rsp is aligned to 8B)

II.Array

1. Array Access

Assume we have an array `int a[100]`, the base address is stored in `%rax`. And we have a variable `long i`, stored in `%rbx`

Please write the instructions:

```
movl    (%rax), %ecx           // Example: %ecx = a[0]
movl    40(%rax), %ecx         // %ecx = a[10]
movl    (%rax, %rbx, 4), %ecx  // %ecx = a[i]
movl    (%rax, %rbx, 8), %ecx  // %ecx = a[i * 2]
leaq    (%rax, %rbx, 4), %rcx  // %rcx = &a[i]
leaq    20(%rax, %rbx, 4), %rcx // %rcx = &a[i + 5]
```

2. Nested Array

- 1) Assume we have two functions:

```
int g(int *p, long i, long j) {
    return p[i * 20 + j - 21];
}
int f(long i, long j) {
    int arr[10][20];
    return g(&arr[1][1], i, j);
}
```

Please fill the blank so that `f` will return `arr[i][j]`;

- 2) Please choose the declarations that **won't** cause a compilation error:

(Note: The answer may be more than one)

- A. `int f(int a[][4][5]);`
- B. `int f(int a[4][][5]);`
- C. `int f(int a[4][5][]);`