计算机系统概论（2024 秋）作业 2 解答

1. 使用不超过 3 条 x86-64 指令实现如下函数:

   其中 x,y,z,w 分别存储于%rdi, %rsi, %rdx, %rcx。返回值存储于%rax。

```
long add(long x, long y,long z,long w) {
    return 32 * x + 8 * y + 4 * z + w;
}
```

   Sol: （注意使用 callee-save 寄存器的情况

```
add:
    leaq    (%rsi, %rdi, 4), %rsi
    leaq    (%rdx, %rsi, 2), %rdx
    leaq    (%rcx, %rdx, 4), %rax
    ret
```

2. 以下给出一个 C 语言函数及其对应的 x86-64 汇编，请对照汇编填充 C 语言代码中缺失的部分。

```
int looper(int n, int val) {
    int i;
    int x = 0;
    for(i = 0; __1__; i++) {
        if( __2__ ) {
            x = __3__;
        } else {
            x += __4__;
        }
    }
    return x;
}
```

Sol1: (1)i < n (2)x + 1 > val (3) x + 2 (4) 1

Sol2: (1)i < n (2)++x <= val (3) x (4) 1

Sol3: (1)i < n (2)x >= val (3) x + 2 (4) 1

Sol4: (1)i < n (2)x < val (3) x + 1 (4) 2

答案不唯一，
注意条件和

+1 +2 对应即可

```
looper:
    movl    $0, %eax
    movl    $0, %edx
.L2
    cmpl    %edi, %edx
    jge     .L4
    addl    $1, %eax
    cmpl    %esi, %eax
    jle     .L3
    addl    $1, %eax
.L3:
    addl    $1, %edx
    jmp .L2
.L4:
    ret
```

3. 以下给出一个 C 语言函数，填写下面汇编代码中缺失的部分：

```
long foo(long val,long n) {
    long i;
    long sum = 0;
    for(i = n - 1; i >= 0 && val < i; --i) {
        sum += i;
    }
    return sum;
}
```

```
foo:
    movq    $0, %rax
    leaq    -1(___1___), %rdx
.L2:
    testq   ___2___, %rdx
    js      .L1
    cmpq    ___3___, ___4___
    jge     .L1
    addq    %rdx, %rax
    subq    $1, %rdx
    jmp     .L2
.L1:
    ret
```

Sol1: (1) %rsi (2) %rdx (3) %rdx (4) %rdi

4. 以下给出一个 C 语言函数:

```
long foo(long a,unsigned long b) {
    long ret = foo(a, b/2);
    return ret * ret * (b % 2 == 1 ? a : 1);
}
```

根据上述函数，填写下面汇编代码中缺失的部分:

```
foo:
    pushq   %rdi
    pushq   __1__
    __2__   %rsi
    call    foo
    __3__    %rax, %rax
    popq    __4__
    popq    __5__
    andq    $1, %rsi
    __6__   __7__, %rsi
    __8__   .L2
    imulq    %rdi, %rax
.L2:
    ret
```

Sol1: (1) %rsi (2) shrq (3) imulq (4) %rsi (5) %rdi (6) testq (7) %rsi (8) je

Sol2: (1) %rsi (2) shrq (3) imulq (4) %rsi (5) %rdi (6) cmpq (7) $1 (8) jne

Sol3: (1) %rsi (2) shrq (3) imulq (4) %rsi (5) %rdi (6) cmpq (7) $0 (8) je

5. 对于下列 x86-32 汇编代码段，从 C 代码中找出其对应的函数实现（填写函数名）

以下是 C 代码:

```
int choice1(int x) { return (x >= 0); }
int choice2(int x) { return ~(x >> 31); }
int choice3(int x) { return 15 * x; }
int choice4(int x) { return (x + 15) / 16; }
int choice5(int x) { return x / 16; }
int choice6(int x) { return 1 - (x >> 31); }
```

以下是需要查找对应 C 代码的汇编代码段

```
foo1:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %eax
    sall    $4, %eax
    subl    8(%ebp), %eax
    movl    %ebp, %esp
    popl    %ebp
    ret
```

Sol for foo1: choice3

```
foo2:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %eax
    testl   %eax, %eax
    jge     .L2
    addl    $15, %eax
.L2:
    sarl    $4, %eax
    movl    %ebp, %esp
    popl    %ebp
    ret
```

Sol for foo2: choice5

```
foo3:
    pushl   %ebp
    movl    %esp, %ebp
    movl    8(%ebp), %eax
    shrl    $31, %eax
    negl    %eax
    addl    $1, %eax
    movl    %ebp, %esp
    popl    %ebp
    ret
```

sol for foo3: choice1

6. X86-64 体系结构中的条件跳转指令 jg 是用于符号数比较还是无符号数比较的?

   其产生跳转的成立条件是 $\sim(SF \oplus OF)\&(\sim ZF)$ 为真，其中 $\oplus$ 表示异或运算，请解释对应标志位的含义并分情况讨论为何是这一条件。

   Sol:

   有符号数比较；jg 是 greater（signed）跳转

   分两种情况讨论：

   1. 当 $OF == 0$ 时，则代表没有溢出，此时 SF 必须为 1，代表结果为负，即 $a < b$。

   2. 当 $OF == 1$ 时，则代表有溢出，此时 SF 必须为 0，即结果最后为正数，那么此时则是负溢出，也可以得到 $a < b$。

   综上，SF != OF 则代表小于的意思，ZF == 1 代表等于，所以 $\sim(SF \oplus OF)\&(\sim ZF)$ 代表大于。

7. 假设定义了以下变量：

```
int x, int y;
unsigned ux = (unsigned)(x);
unsigned uy = (unsigned)(y);

float f;
double d;
assert(!isnan(f) && !isnan(d)); // 保证 f 和 d 都不是 NaN
```

   判断下列逻辑表达式在 C++ 中运行的结果是否永远为真，若可能为假请解释或给出反例。

```
x == (int)(double)x
ux == x
x + uy == y + ux
(x > 0) || (-x >= 0)
(x >> 4) == x / 16
(ux >> 4) == ux / 16
((x | ~x) >> 31) == -1
((x & -x) != 0) || (x == 0)
(d + f) - d == f
```

   Sol:

```
True
True
True
False   x = INT_MIN
False   x = -1
True
True
True
False   浮点运算
```