**1.** 假设寄存器大小为8位

两个十进制数 a=6,b=-1,则a-b的标志位 ZF=\_\_ SF=\_\_ CF=\_\_\_ OF=\_\_\_

a=6,b=-5, 则a-b的标志位 ZF=\_\_ SF=\_\_ CF=\_\_\_ OF=\_\_\_

**2.** 在下列指令中，其执行会影响条件码中的CF位的是：

A. jmp NEXT B. jmp NEXT C. inc %bx D. shl $1,%ax

**3.** 下列指令中不会改变条件码寄存器内容的是 。

A CMP B TEST C ADD D LEA

**4.** 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?

A 1)和2)都不会 B 1) C 2) D 1)和2)都会

**5.** 下列指令中不会改变PC寄存器内容的是

A ADD B JMP C CALL D RET

**6. 下面哪条指令不会引起esp的变化？**

* 1. **movl %esp, %ebp**
  2. **pushl %ebp**
  3. **call printf**
  4. **subl $20, %esp**

**7. 下面哪条指令不是X86正确的寻址方式**

1. **movl $34, (%eax)**
2. **movl (%eax), %eax**
3. **movl $23, 10(%edx, %eax)**
4. **movl (%eax), 8(%ebx)**

**8.** 下列关于比较指令CMP说法中，正确的是：

A. 专用于有符号数比较 B. 专用于无符号数比较 C. 专用于串比较 D. 不区分比较的对象是有符号数还是无符号数

**9.** 将AX清零，下列指令错误的是（ ）

A. sub %ax, %ax B. xor %ax, %ax

C. test %ax, %ax D. and $0, %ax

**10.** 已知短整型数组S的起始地址和下标i分别存放在寄存器%rdx和%rcx，将 &S[i]存放在寄存器%rax中所对应的汇编代码是（ ）

A. leaq (%rdx, %rcx, 1), %rax B.movw (%rdx, %rcx, 2), %rax

C. leaq (%rdx, %rcx, 2), %rax D.movw (%rdx, %rcx, 1), %rax

**11.** 跳转表是连续的，题目编排有点小问题

标号没有出现分支表示默认跳转

表格

AI 生成的内容可能不正确。

**12.** 定义结构struct S1 {char c; int i; char d;} \*p;，其所占用的内存空间为 \_\_\_个字节。 A 6 B 7 C 9 D 12  
**13.**

图片包含 散点图

AI 生成的内容可能不正确。

**14.**

文本

AI 生成的内容可能不正确。

**15.**图形用户界面, 文本

AI 生成的内容可能不正确。

**16.**

对于int b[10][5]，有如下代码

for(i=0;i<10;i++)

for(j=0;j<5;j++)

sum+=b[i][j]

假设执行到sum+=…时，sum在eax中，b[i][0]所在的地址在rdx，j在esi，则 sum+=b[i][j]

对应的指令可以是( )

（a）addl 0(%rdx,%esi,4),%eax

（b）addl 0(%esi,%rdx,4),%eax

（c）addl 0(%rdx,%esi,2),%eax

（d）addl 0(%esi,%rdx,2),%eax

**17.**

IA-32 指令，pushl %ebp 对应的指令（ ）：

（a）R[esp]=R[esp]-4,M(R[esp]) =R[ebp]

（b）R[esp]=R[esp]+4,M(R[esp]) =R[ebp]

（c）M(R[esp]) =R[ebp] ,R[esp]=R[esp]-4

（d）M(R[esp]) =R[ebp] ,R[esp]=R[esp]+4

**18.**

文本

AI 生成的内容可能不正确。

**19.**

图片包含 图形用户界面

AI 生成的内容可能不正确。

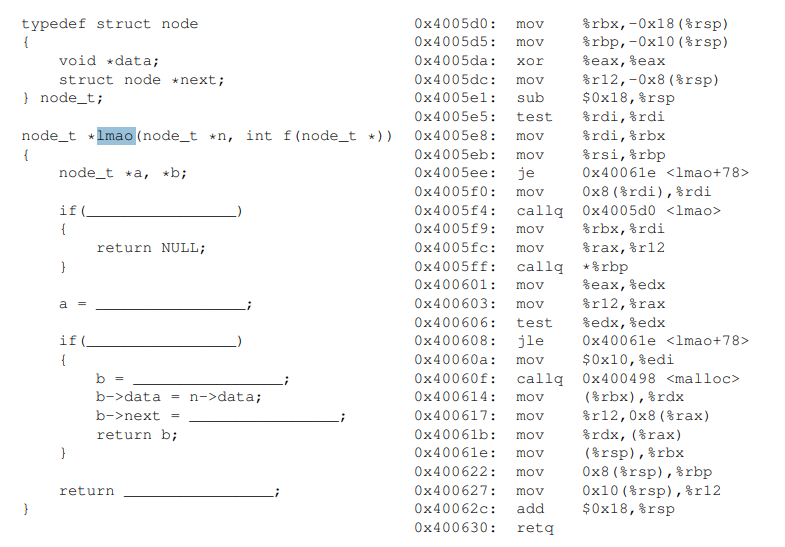
**20.**

手机屏幕截图

AI 生成的内容可能不正确。

**21.**

用适当的表达式填充lmao的C代码中缺失的部分。（注：0x400498是C标准库函数malloc的地址。）

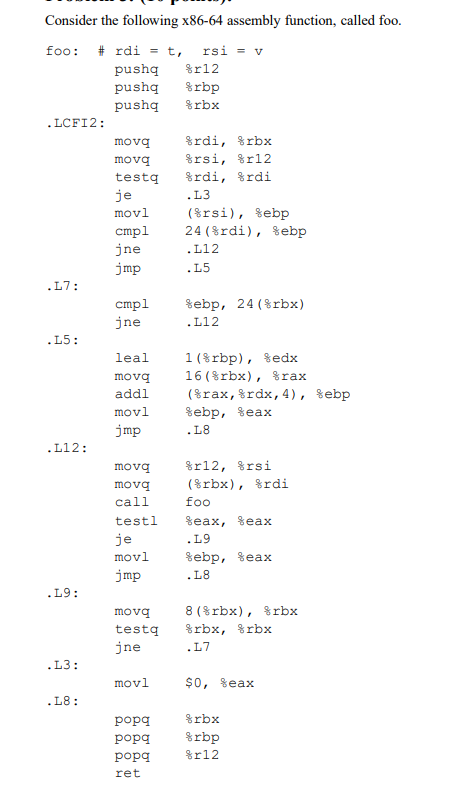


**22.**

文本, 表格

AI 生成的内容可能不正确。

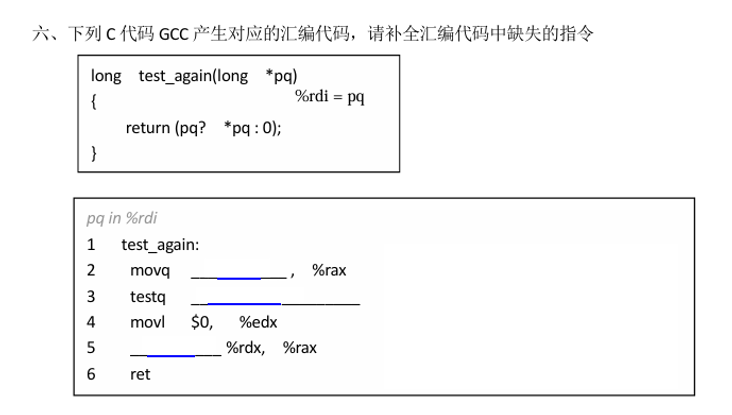
**23.**



文本

AI 生成的内容可能不正确。

**24.**



**25.**

文本

AI 生成的内容可能不正确。

**26.**

文本, 信件

AI 生成的内容可能不正确。

**27.**

文本

AI 生成的内容可能不正确。

**28.**

文本

AI 生成的内容可能不正确。

**29.**

表格

AI 生成的内容可能不正确。1

**30.**

文本

AI 生成的内容可能不正确。

**31.**

struct mystruct {

char c;

double\* dp;

int i;

} m[2];

假设m存放的地址是0x100

请问下面语句的输出是什么？ printf(“%x %x %x \n”, &m[0].dp, m+1, &m[1].i);

**32.**

文本

AI 生成的内容可能不正确。

**33.**

文本

AI 生成的内容可能不正确。

**34.**

1. （堆栈破坏问题）函数echo定义如下：

void echo(){

char buf[8];

gets(buf);

puts(buf)；

返回地址（64位）

echo

的栈帧

%rsp

echo

的栈帧

%rsp

}

对应的汇编代码如下：

echo:

subq $24,%rsp

movq %fs:40,%rax

movq %rax,8(%rsp)

xorl %eax,%eax

movq %rsp,%rdi

call gets

movq %rsp,%rdi

call puts

movq 8(%rsp),%rax

xorq %fs:40,%rax

je .L9

call \_\_stack\_chk\_fail

addq $24,%rsp

ret

观察代码，判定该函数是否具有堆栈破坏的检测能力？

如果%fs:38地址开始存放0x00/01/02/03/04/05/06/07/08/09/0a/0b/0c/0d/0e/0f。请问刚进入echo函数时，echo栈帧中%rsp位置存放的8字节数值是？

如果此时输入按键abcdefg并回车，程序将如何执行？

如果此时输入按键123456789并回车，程序能否正常返回？如果不能将执行什么处理？

**35.**

已知函数loop的C语言代码框架及其过程体对应的汇编代码，根据对应的汇编代码填写C代码中缺失的语句或表达式。

loop:

# on entry: a in %rdi, n in %esi

movl $0, %r8d

movl $0, %ecx

testl %esi, %esi

jle .L3

.L6:

movl (%rdi,%rcx,4), %edx

leal 3(%rdx), %eax

testl %edx, %edx

cmovns %edx, %eax

sarl $2, %eax

addl %eax, %r8d

addq $1, %rcx

cmpl %ecx, %esi

jg .L6

.L3:

movl %r8d, %eax

ret

注意：只能使用C语言代码中的变量（包括n、a、i和sum）

int loop (int a[ ], int n)

{

int i, sum;

sum = \_\_\_\_\_;

for (i = \_\_\_\_\_\_\_\_\_\_\_\_; \_\_\_\_\_\_\_\_\_\_\_\_; \_\_\_\_\_\_\_\_\_\_\_\_) {

sum += \_\_\_\_\_\_\_\_\_\_\_\_\_\_;

}

return \_\_\_\_\_\_\_\_\_\_\_\_;

}

4、函数func定义如下，请完成func函数对应的汇编代码中的空缺

void func (int \*sum)

{

int arr[4] = {2, 0, 1, 9};

for ( int i = 0; i < 4; i++ )

\*sum += arr[i];

}

func:

subq $0x10, %rsp #

movl $0x09, 12(%rsp) #

#

movl $0x2, (%rsp) #

movl $0, %eax

movl $0, %ecx

.L1

jl .L1

addl %ecx, (%rdi) #

ret

**36.**

对于以下代码：

int a=100;

short proc(int inputarg)

{ int a;

a+=inputarg;

return a+0;

}

请说明边第1行变量a和3行定义变量a的作用域。如果将第3行变量类型修改为static int，则第一次调用函数proc时传入参数inputarg的值为12，此时返回值为多少？再次调用时传入inputarg为5，则第二次调用的返回值是多少？

**37.**

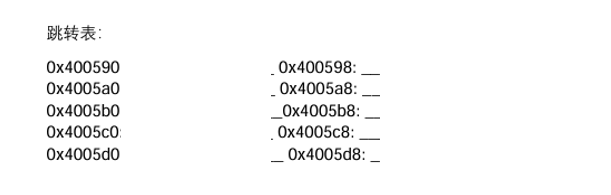
图形用户界面, 文本, 应用程序

AI 生成的内容可能不正确。

**38.**

手机屏幕截图

AI 生成的内容可能不正确。



**39.**

文本

AI 生成的内容可能不正确。

40.

考虑以下C函数调用和对应的x86-64汇编代码片段：

void func(int a, int b, int c, int d, int e, int f, **int g, int h**) {

int local1 = a + b;

int local2 = c + d;

int local3 = e + f;

int local4 = g + h;

// ... 其他操作

}

int main() {

func(1, 2, 3, 4, 5, 6, 7, 8);

return 0;

}

对应的x86-64汇编（部分）如下：

func:

pushq %rbp

movq %rsp, %rbp

subq $32, %rsp

movl %edi, -4(%rbp) ; a

movl %esi, -8(%rbp) ; b

movl %edx, -12(%rbp) ; c

movl %ecx, -16(%rbp) ; d

movl %r8d, -20(%rbp) ; e

movl %r9d, -24(%rbp) ; f

movl 16(%rbp), %eax ; g

movl %eax, -28(%rbp)

movl 24(%rbp), %eax ; h

movl %eax, -32(%rbp)

; ... 计算局部变量

‌问题：‌

1）‌栈帧布局‌：

* 在func的栈帧中，参数g和h的存储位置相对于%rbp的偏移量是多少？为什么？
* 为什么subq $32, %rsp为局部变量预留了32字节空间？

‌2）调用前后栈指针变化‌：

* 在main调用func之前，栈指针%rsp需要如何调整？
* 画出调用func时的完整栈帧结构（包括返回地址、保存的%rbp、参数和局部变量）。