**1，**考虑以下在x86-64系统上定义的结构体和C函数代码，

struct my\_struct {

char a;

long b;

short c;

float \*d[2];

unsigned char e[3];

float f;  
};

void foo(struct my\_struct \*st) {

st->a = ’e’;

st->d[0] = NULL;

st->c = 0x213;

printf("%lld %p %hhu\n", st->b, &st->f, st->e[1]);

}

请回答如下问题：

1. 对齐方式下，该结构体需要多少个字节？**请画图说明每个结构体成员的存储位置**，偏移量分别是多少（以字节为单位）

48字节， 偏移为： 0， 8， 16， 24， 40， 44,

（2）根据foo函数代码，填写以下缺失的汇编代码中。

0x00000000004004e4 <+0>: sub $0x8, %rsp

0x00000000004004e8 <+4>: movb $0x65，(%rdi)

0x00000000004004eb <+7>: movq $0x0,\_\_\_24\_\_(%rdi)

0x00000000004004f3 <+15>: movw $0x213,\_\_16\_\_\_(%rdi)

0x00000000004004f9 <+21>: movzbl \_\_\_41\_\_(%rdi), %ecx

0x00000000004004fd <+25>: lea \_\_\_\_44\_\_ (%rdi), %rdx

0x0000000000400501 <+29>: mov \_\_\_\_8 \_(%rdi), %rsi

0x0000000000400505 <+33>: mov $0x40062c, %edi

0x000000000040050a <+38>: mov $0x0, %eax

0x000000000040050f <+43>: callq 0x4003e0 <printf@plt>

0x0000000000400514 <+48>: add $0x8, %rsp

0x0000000000400518 <+52>: retq

1. 该结构体经过优化后，对齐方式下最少需要多少个字节可以存储？

40个字节

1. 具有一个通用结构的C函数如下(部分代码缺失)

long rfun(unsigned long x) {

if ( x==0 )

return 0 ;

unsigned long nx= x/4 ;

long rv = rfun(nx);

return x+rv ;

}

GCC对其参数如下汇编代码：

rfun:

pushq %rbx

movq %rdi, %rbx

movl $0, %eax

testq %rdi, %rdi

je .L2

shrq $2, %rdi

call rfun

addq %rbx, %rax

.L2:

popq %rbx

ret

1）根据上面的汇编代码补全rfun函数中的C代码。

2）记指令 addq %rbx, %rax 的地址为address，假设函数的入口参数x=4，画出函数从执行rfun(4)到执行rfun(0)（执行rfun(0)的第一条指令前）栈帧存放的数据的情况（存放的值及位置）。

%rbx

Address

%rbx=4

address

3 已知函数rec的C语言代码，请补齐缺失的x86-64汇编代码，并注解关键的指令含义。

long rec (int n)

{

if (n >1)

return n \* rec(n-1);

else

return 1;

}

rec:

pushq %rbx

Movq %rdi, %rbx //

movl $1, %eax

subl $1, %rdi //

Jle .L1 //

call rec //

imulq %rbx, %rax

.L1:

Popq %rbx //

ret

4 将源程序INC.c通过如下命令编译并形成静态库libINC.a

$gcc –c INC.c

$ar rcs libINC.a INC.o

1. 基于以上步骤获得的静态库，请问使用如下命令能否成功执行？请给出解释。

$gcc –o main ./libINC.a main.c

不能， 无法完成符号INC的解析

1. 请问使用如下命令能否成功编译main.c？请给出解释。

$gcc –o main main.c ./libINC.a

不能， 有两次对sum的强定义，链接会报错

|  |
| --- |
| /\* INC.c \*/  #include <stdio.h>  double sum =1.0;  void INC(void)  {  sum = sum + 1.0;  printf(“sum = %lf\n”, sum);  } |

|  |
| --- |
| /\* main.c \*/  #include <stdio.h>  void INC(void);  int sum =0;  int index = 1;  int main(int argc, char \*\*argv)  {  INC();  printf(“sum = %d\n”, sum);  } |