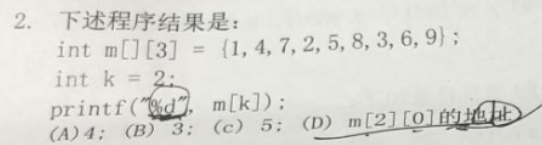
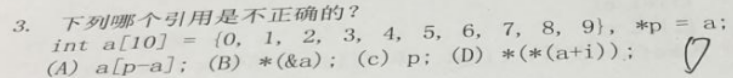


10 9 4 9



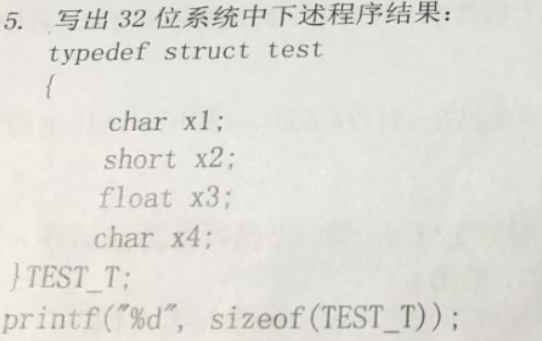
(D)



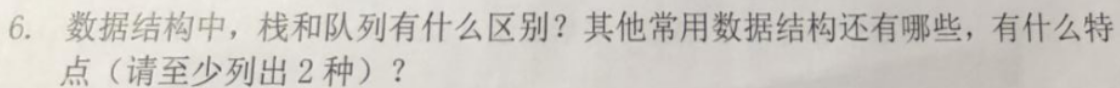
(D)



#define MIN(A,B) (A)<(B)?(A):(B)



12



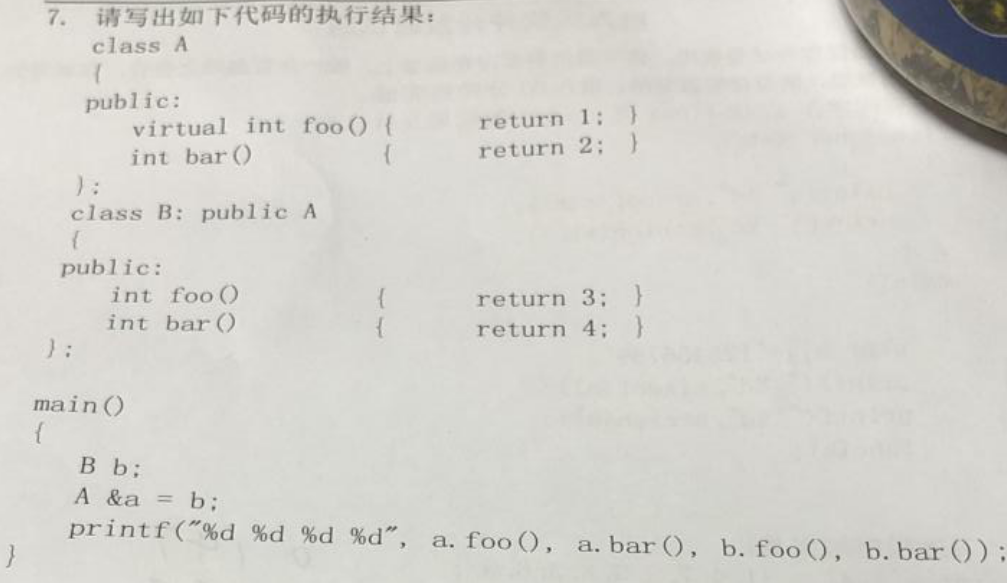
栈：先进后出

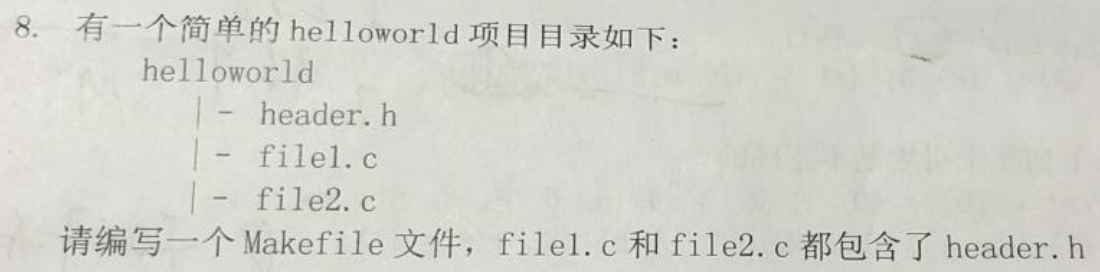
队列：后进先出

其他常用数据结构：顺序表，链表，树，图

树：一对多

图：多对一





CC=gcc

CFLAGS=-c -g -O

OBJS=file1.o file2.o

a.out:$(OBJS)

$(CC) $(OBJS) -o $@

$(OBJS):%.o:%.c

$(CC) $(CFLAGS) $^ -o $@

.PHONY:clean

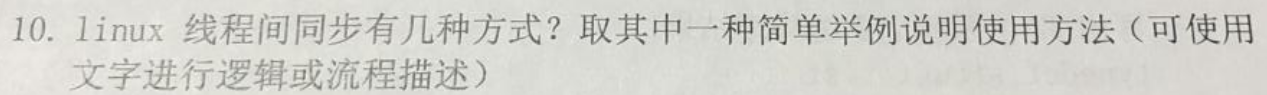
clean:

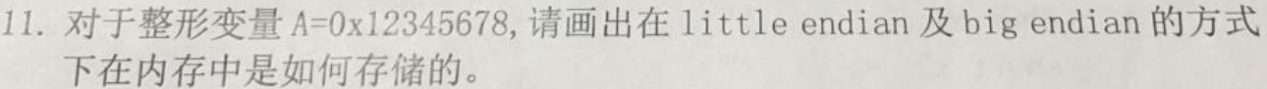
rm \*.o add



线程共享本进程的资源如内存、I/O、cpu等，不利于资源的管理和保护，而进程之间的资源是独立的，能很好的进行资源管理和保护；

线程共享本进程的地址空间，而进程之间是独立的地址空间。





|  |  |  |  |
| --- | --- | --- | --- |
| 12 | 34 | 56 | 78 |

大端：

低地址 高地址

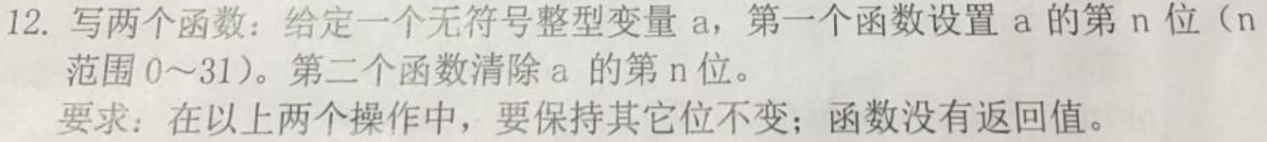
|  |  |  |  |
| --- | --- | --- | --- |
| 78 | 56 | 34 | 12 |

小端：

低地址 高地址

大端：低地址存放高字节数据

小端：低地址存放低字节数据



void shezhi(unsigned int a, int n,int flag)

//置位→ num:无符号整型a，n:从右往左第n位，flag:0或1

{

if(flag==1){

a = a| 1<<(n-1);

}

else

a = num & (~(1<<(n-1)));

printf("%u\n",a);

}

void qingchu(unsigned int a, int n) //清除→a:无符号整型a，n:从右往左要清除的第n位

{

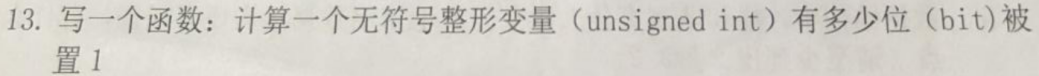
int cp = a & ((1<<(n-1))-1);

a = a >> n;

num = (a<<(n-1)) | cp;

printf("%u\n", a);

}



int func(unsigned int date)

{

int count = 0;

while(date)

{

if(date & 1 == 1)

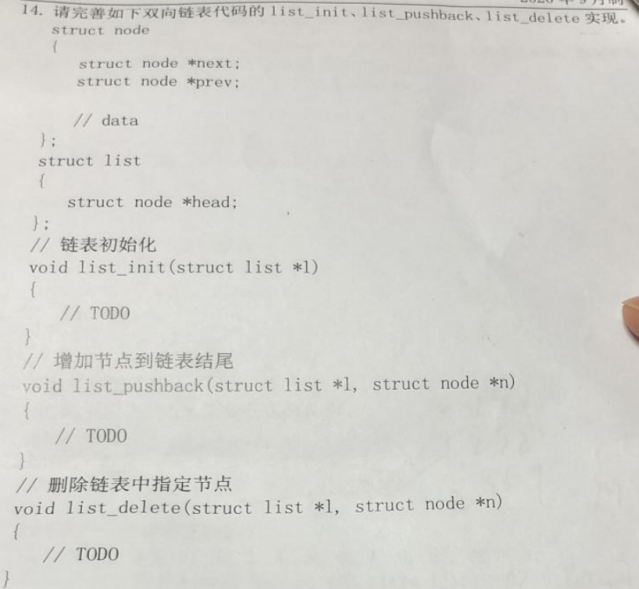
count++;

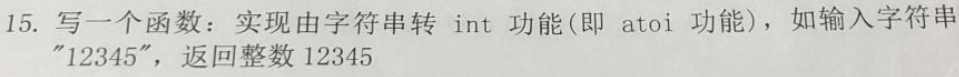
date = date>>1;

}

return count;

}





int atoi(char \*str)

{

int num = 0;

while(\*str)

{

num = num\*10 + (\*str-48);

str++;

}

return num;

}