两套,题型分值分布如下

一、 填空题（每空2分，共20分）

二、 判断题（正确√，错误×）（每题2分，共20分）

三、 简答题(每小题5分，共20分)

四、 程序分析题。请阅读程序并写出运行结果（每小题10分，共20分）。

五、 程序题。（共20分）

一、 填空题（每空2分，共20分）

二、 判断题（正确√，错误×，将答案填写到下面的表格中）（每题2分，共24分）

三、 简答题(每小题5分，共20分)

四、 程序分析题。请阅读程序并写出运行结果（每小题10分，共20分）。

五、 程序题。（共16分）

一、填空题（每空 2 分，共 20 分）

1、网络编程中的 API 是指_____。

2、Socket 函数中，对应 TCP 四层协议中的控制层，有两个主要协议名称分别为：_____、_____；其协议参数分别为：_____、_____。

3、并发设计中的主要两种运行单元分别是 _____、_____。

4、主进程等待子进程结束可使用的函数_____。

5、DNS 主要可提供对_____和_____的查询服务。

1、Socket 编程可支持对 IP 报头的自组包；

2、UDP 编程必须使用 bind 函数；

3、Socket 函数中，IPv4 和 IPv4 都是使用 INET 参数作为协议名；

4、若父进程要等待子线程结束，则子线程创建时不能设置 detach 属性；

5、signal 函数可以设定自定义的信号响应处理过程；

6、select 函数可以作为定时函数使用；

7、exec 函数与 fork 函数的执行行为是一致的；

8、一个完整的网络连接应包含本地 IP，本地 Port，以及对等端 IP，对等端 Port 的信息；

9、可以用 socket 编程方法或者修改系统文件方法设定系统最大的监听队列长度；

10、守护进程须关闭控制终端；

11、服务端崩溃时，会给对等端发送 RST 报文；

12、TCP 和 UDP 不能使用相同端口号。

三、简答题（每小题 5 分，共 20 分）

1、使用 TCP 通信模型描述服务器端的 socket 编程过程；

2、给出通信双方进行套接字通信的基本信息(5 元组)。

3、I/O 复用中可控制的三种集合类型；

4、给出三种为新线程传递参数的方法。

2、给出下面函数的作用：

struct sockaddress saddr;

（1）bezro(&saddr, sizeof(saddr));

（2）saddr.sin_family = AF_INET;

（3）saddr.sin_port = htons(3000);

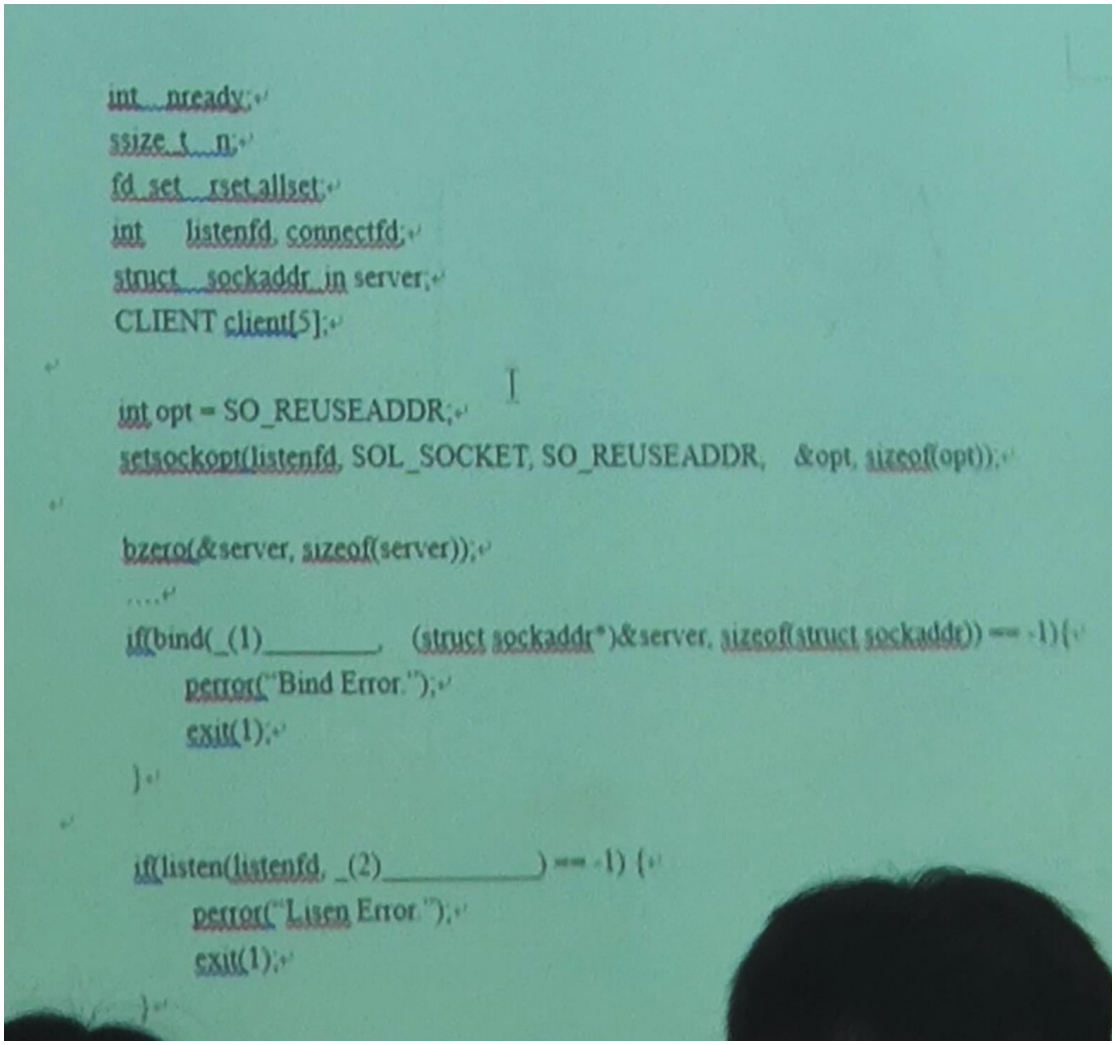
（4）saddr.sin_addr.s_addr = htonl(INADDR_ANY);

（5）perror("address Error");

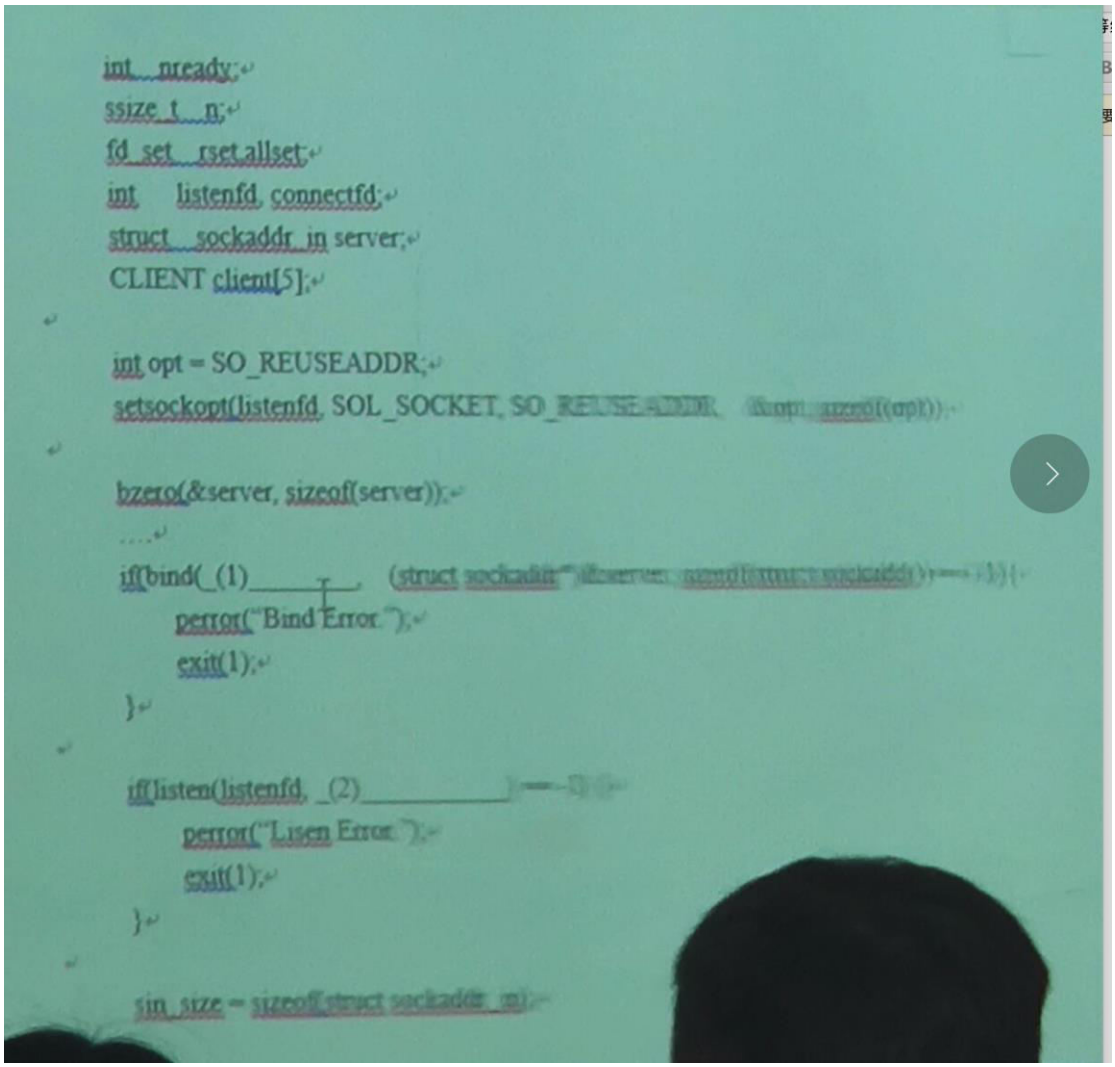
五、程序题。（共 16 分）

本题主要考核 I/O 编程中 FD_**宏定义函数对读、写、异常集的操作。根据要求，阅读程序，在程序的画横线的空白处补全代码。

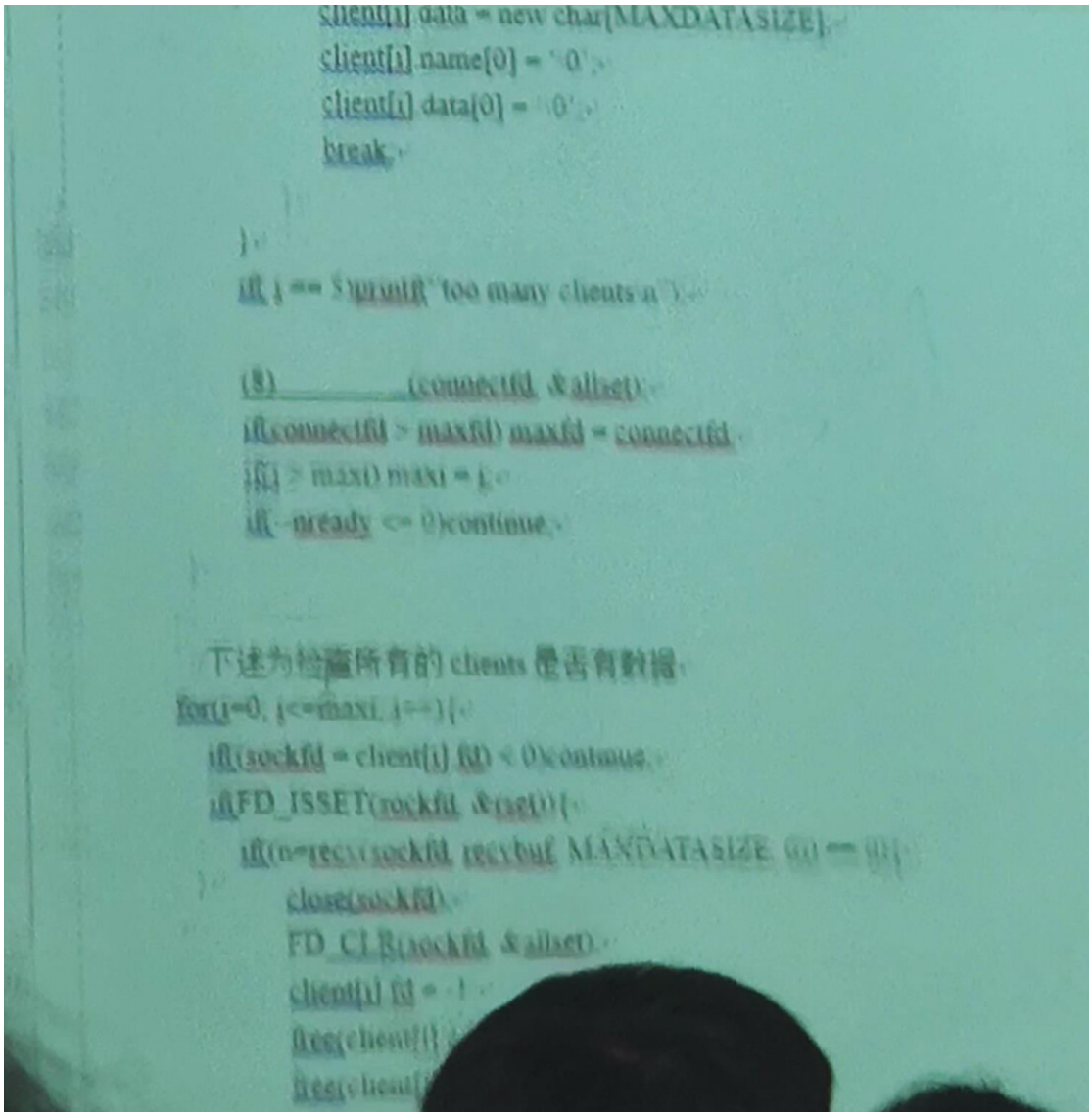
```
#define PORT 1234
#define BACKLOG 5
#define MAXDATASIZE 1000
typedef struct CLIENT{
    int fd;
    char* name;
    struct sockaddr_in addr;
    char* data;
};

main()
{
    int  i, maxi, maxfd, sockfd;
```

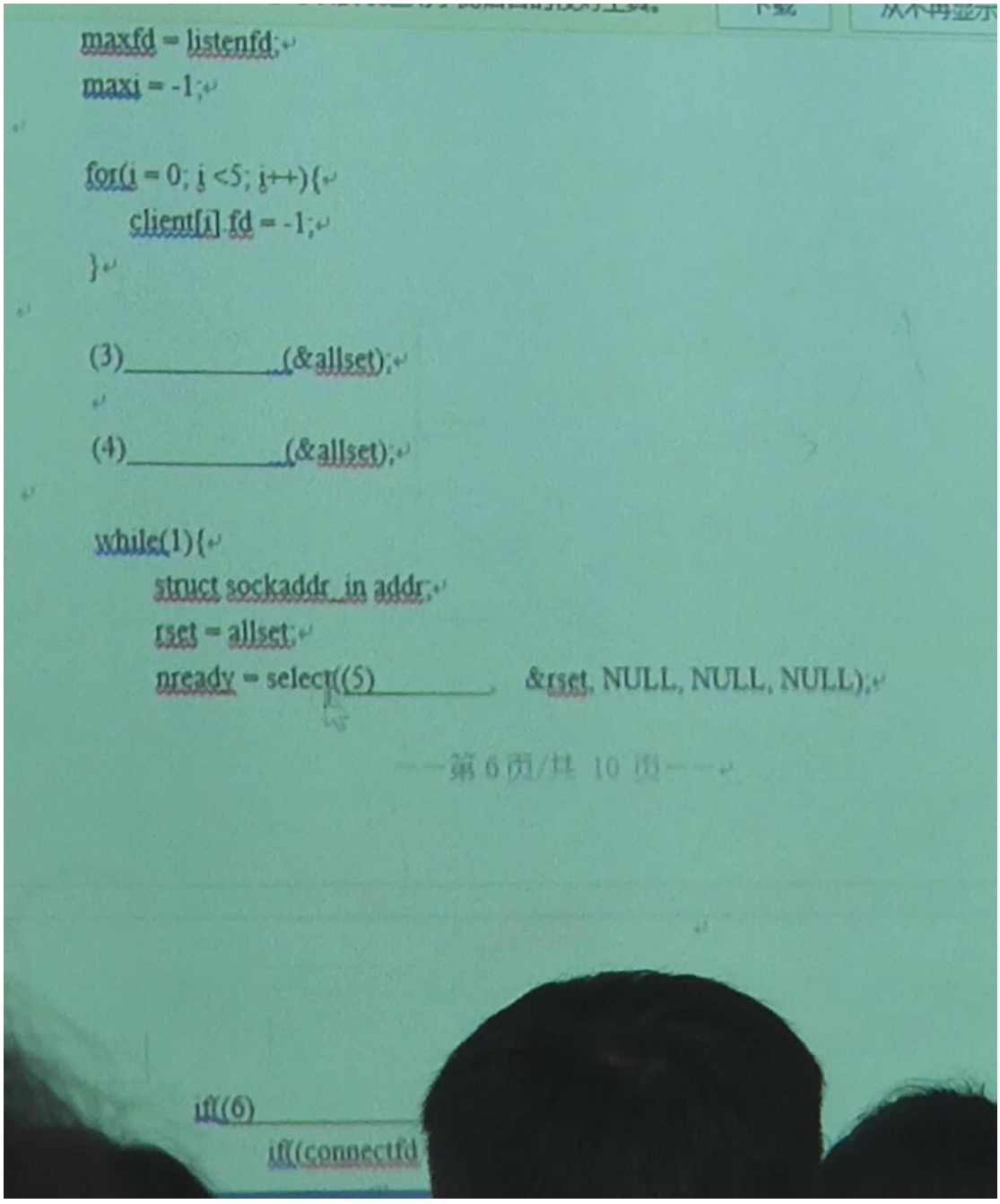


```
int   nready;
ssize_t   n;
fd_set   rset,allset;
int     listenfd, connectfd;
struct   sockaddr_in server;
CLIENT client[5];

int opt = SO_REUSEADDR;
setsockopt(listenfd, SOL_SOCKET, SO_REUSEADDR,   &opt, sizeof(opt));

bzero(&server, sizeof(server));
....
if(bind(_(1)_____,   (struct sockaddr*)&server, sizeof(struct sockaddr)) == -1){
      perror("Bind Error.");
      exit(1);
}

if(listen(listenfd, _(2)_____) == -1) {
      perror("Lisen Error.");
      exit(1);
}
```



```
int  nready;
ssize_t  n;
fd_set  rset,allset;
int    listenfd, connectfd;
struct  sockaddr_in server;
CLIENT client[5];

int opt = SO_REUSEADDR;
setsockopt(listenfd, SOL_SOCKET, SO_REUSEADDR,  (&opt, sizeof(opt));

bzero(&server, sizeof(server));
...
if(bind(_(1)_____,  (struct sockaddr*)&server, sizeof(struct sockaddr)) == -1){
    perror("Bind Error.");
    exit(1);
}

if(listen(listenfd, _(2)_____) == 0){
    perror("Lisen Error.");
    exit(1);
}

sin_size = sizeof(struct sockaddr_in);
```

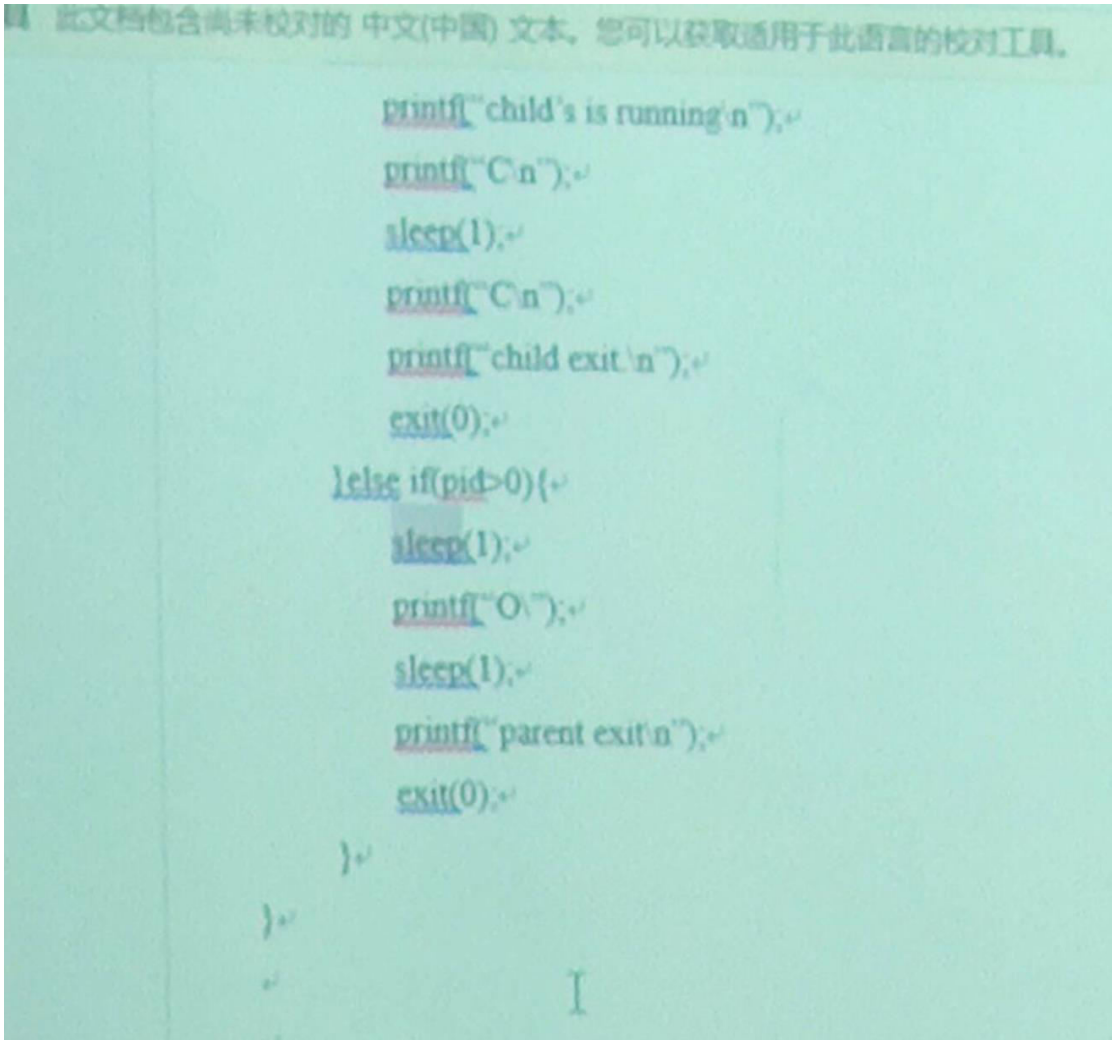




```
maxfd = listenfd;
maxi = -1;

for(i = 0; i <5; i++){
    client[i].fd = -1;
}

(3)_____(&allset);

(4)_____(&allset);

while(1){
    struct sockaddr_in addr;
    rset = allset;
    nready = select((5)_____, &rset, NULL, NULL, NULL);
```



```
    if((6)_____
        if((connectfd
```



就是类似于复习课老师讲的那个　　多线程 运行一下就回来，所以要注意延迟时间