得分

第七题（10 分）

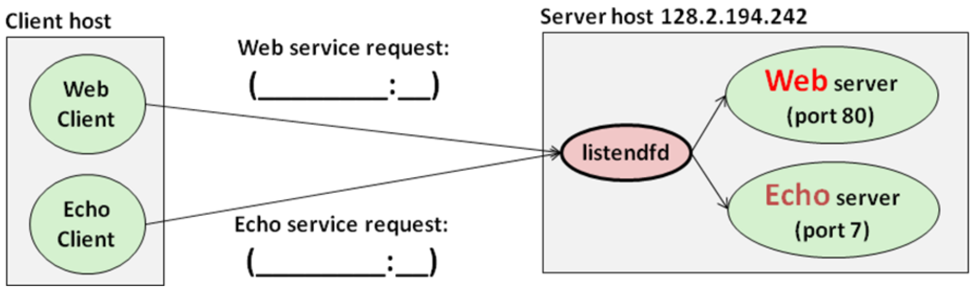
（1） 一个服务器拥有两个独立的固定 IP 地址，那么它在 web 应用端口 80 上最多可以监听多少个独立的 socket 连接？（2 分）

服务器端	客户端	结果

（2）该服务器在所有有 web 应用端口上最多可以监听多少个独立的 socket 连接？（2 分）

服务器端	客户端	结果

（3） 在下图中连线上填入正确的目标服务器的 socket 标识符（2 分）



4) 在 Echo server 范例中，server 端通过 accept 函数接受了一个 client 的连接请求，从而将网络描述符与该网络连接、socket 绑定，然后进行网络数据传输。在下面的空格处填写正确的网络描述符，每个空填写 **listenfd** 或 **connfd**（共 4 分，每空 1 分）。

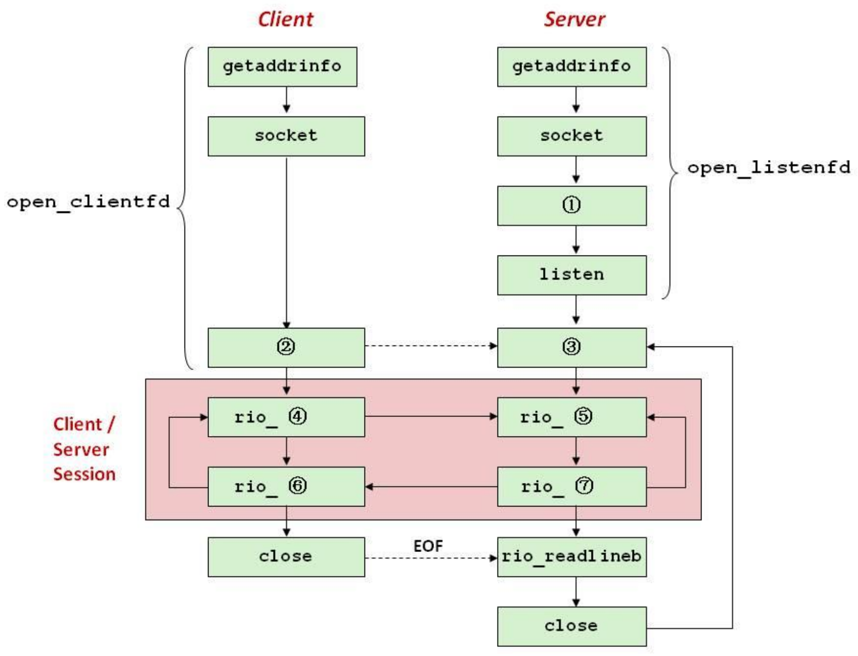
```
int main(int argc, char **argv) {
   int listenfd, connfd, port, clientlen;
   struct sockaddr_in clientaddr;
   struct hostent *hp;
   char *haddrp;
   unsigned short client_port;
     …………
     while (1) {
        clientlen = sizeof(clientaddr);
        _________ = Accept(_________, (SA *)&clientaddr,
                  &clientlen);
        hp = Gethostbyaddr((const char*)
            &clientaddr.sin_addr.s_addr,
            sizeof(clientaddr.sin_addr.s_addr), AF_INET);
        haddrp = inet_ntoa(clientaddr.sin_addr);
        client_port = ntohs(clientaddr.sin_port);
        printf("server connected to %s (%s), port %u\n",
                 hp->h_name, haddrp, client_port);
        echo(_________);
        Close(_________);
        }
     }
```

得分

第六题（10 分）

下图是一个基于 echo 服务器的 client-server 框架

（1）　请给图中的编号填写相应的函数名。



（2）　请补全下面 server 端 open_listenfd 函数中缺失的操作（Line 21, Line 26 和 Line 34）

```
Line 1:  int open_listenfd(char *port)
Line 2:  {
Line 3:     struct addrinfo hints, *listp, *p;
Line 4:     int listenfd, optval=1;
Line 5:     /* Get a list of potential server addresses */
Line 6:     memset(&hints, 0, sizeof(struct addrinfo));
Line 7:     hints.ai_socktype = SOCK_STREAM;
Line 8:     hints.ai_flags = AI_PASSIVE | AI_ADDRCONFIG;
Line 9:     hints.ai_flags |= AI_NUMERICSERV;
Line 10:    Getaddrinfo(NULL, port, &hints, &listp);
Line 11:
Line 12:    for (p = listp; p; p = p->ai_next) {
Line 13:        /* Create a socket descriptor */
Line 14:       if ((listenfd = socket(p->ai_family, p->ai_socktype,
Line 15:                              p->ai_protocol)) < 0)
Line 16:          continue;  /* Socket failed, try the next */
Line 17:         /* Eliminates "Address already in use" error from
bind */
Line 18:        Setsockopt(listenfd, SOL_SOCKET, SO_REUSEADDR,
Line 19:                   (const void *)&optval , sizeof(int));
Line 20:
Line 21:        if (_⑧_(listenfd, p->ai_addr, p->ai_addrlen) == 0)
Line 22:          break;  /* Success */
Line 23:        Close(listenfd);
Line 24:    }
Line 25:    /* Clean up */
Line 26:    Freeaddrinfo(___⑨___);
Line 27:    if (!p) /* No address worked */
Line 28:        return -1;
Line 29:
Line 30:    if (listen(listenfd, LISTENQ) < 0) {
Line 31:        Close(listenfd);
Line 32:        return -1;
Line 33:    }
Line 34:    return __⑩___;
Line 35: }
```