AD_OS 实验报告 实验一 董天智 2017100937

一、实验题目

UNIX/Linux 环境下客户/服务器网络编程：

1、 单机程序网络版——掷骰子

2、 基于 socket()的 TCP 的分布式文件系统

二、实验目标

1、 了解单机程序与网络版程序的不同处

2、 熟悉客户/服务器编程模式

3、 熟悉 python 下的 socket 网络编程，包括建立、绑定、监听、连接、关闭等

4、 熟悉文件处理

5、 巩固 python 命令行的参数传递

三、实验内容

1、 单机程序网络版——掷骰子

首先实现一个单机版的掷骰子程序。然后改为采用客户/服务器模式，由客户端发起请求，服务器接受到客户端发来的特定命令后返回一个 1-6 的随机整数，客户端接受到服务器的结果后显示到标准输出。

2、 基于 socket()的 TCP 的分布式文件系统

实现一个分布式文件系统，客户端可以向服务器上传、下载文件，也可以获取服务器上存储的文件列表。

四、实验原理与算法

1、 C/S 模式

客户端和服务器端模式是指在服务器端部署一个程序，在客户端也部署一个程序，服务器端的程序负责相应客户端发来的各种请求，处理后给客户端返回相应的结果。客户端用于和用户交互以及发送请求到服务器和接受服务器的相应。

2、 Socket 网络编程

Socket 是一种面向连接的通信方式，通信双方分为客户和服务器两个角色，服务器绑定本地的某个端口并监听外部的连接。客户端通过 ip 地址和端口制定要连接的服务器。

3、 文件操作

文件操作在 python 中比较方便，主要是调用 open 函数，以及制定读写方式和编码。

五、伪码算法

1、 单机程序网络版——掷骰子

```
单机版：
While(True):
    用户输入命令；
    生成 1-6 的随机数；
    显示生成的随机数；
网络版-服务器：
创建套接字，启动监听，等待连接；
While(True):
        接收到客户端请求，建立连接；
        生成随机数；
        发送生成的随机数给客户端；
    网络版-客户端：
    创建套接字，和服务器建立连接；
    While（True）：
        接受用户的指令；
        发送指令到服务器；
        接受服务器发来的随机数；
        把接受到的随机数展示给用户；
```

2、 基于 socket()的 TCP 的分布式文件系统

```
    服务器：
    创建套接字，启动监听，等待连接；
    接收连接；
    While（True）：
        客户端发来的命令；
        根据命令种类执行不同的操作（upload，download，list）
        返回执行结果给客户端；
    客户端：
    创建套接字，和服务器建立连接；
    While（True）：
        提示用户输入想执行的操作；
        发送操作命令到服务器；
        接受来自服务器的相应；
        显示执行结果
```

六、程序源码

1、 单机程序网络版-掷骰子

```
import socket
import sys
import random as rd

def stand_alone():
    while True:
        print('请投骰子(输入"go")')
        cmd = str(sys.stdin.readline()).strip('\n')
        if cmd == 'exit':
            break
        elif cmd == 'go':
            reply = str(rd.randint(1, 6))
        else:
            reply = 'command not defined: ' + cmd

        print('骰子点数：  ' + reply)
        print('')
```

```
def client():
        host = 'localhost'
        port = 8888

        try:
                # create an AF_INET, STREAM socket (TCP)
                s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                print('Socket Created')
        except socket.error as msg:
                print('Failed to create socket. Error code: ' + str(msg[0]) + ' , Error message : ' +
msg[1])
                sys.exit()

        remote_ip = socket.gethostbyname(host)
        print(remote_ip)
        print(port)
        s.connect((remote_ip, port))
        print('Socket Connected to ' + host + ' on ip ' + remote_ip)

        while True:
                try:
                        print('请投骰子(输入"go")')
                        cmd = str(sys.stdin.readline()).strip('\n')
                        if cmd == 'exit':
                                break

                        # Set the whole string
                        s.sendall(bytes(cmd, encoding='utf-8'))
                        s.send()
                        # print('Message send successfully')

                        # Now receive data
                        reply = s.recv(4096)
                        s.sendfile()
                        print('骰子点数：   ' + str(reply, encoding='utf-8').strip())
                        print('')
                except socket.error:
                        # Send failed
                        print('Send failed')
                        sys.exit()

        s.close()
```

```
def server():
    HOST = ''    # Symbolic name meaning all available interfaces
    PORT = 8888    # Arbitrary non-privileged port

    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        print('Socket created')
    except socket.error as msg:
        print('Failed to create socket. Error code: ' + str(msg[0]) + ' , Error message : ' +
msg[1])
        sys.exit()

    try:
        s.bind((HOST, PORT))
        print('Socket bind complete')
    except socket.error as msg:
        print('Bind failed. Error Code : ' + str(msg[0]) + ' Message ' + msg[1])
        sys.exit()

    s.listen(10)
    print('Socket now listening on port: ' + str(PORT))

    conn, addr = s.accept()
    print('Connected with ' + addr[0] + ':' + str(addr[1]))

    # now keep talking with the client
    while True:
        # wait to accept a connection - blocking call
        data = conn.recv(4096)

        if not data:
            print('Disconnected with ' + addr[0] + ':' + str(addr[1]))
            break

        cmd_recv = str(data, encoding='utf-8').strip()
        if cmd_recv == 'go':
            reply = str(rd.randint(1, 6))
        else:
            reply = 'command not defined: ' + cmd_recv

        conn.sendall(bytes(reply, encoding='utf-8'))
        print('replt to client: ' + reply)

    conn.close()
    s.close()


if __name__ == '__main__':
    if len(sys.argv) == 1:
        print('role not specified(server or client)')
        exit()
    if sys.argv[1] == 'server':
        server()
    elif sys.argv[1] == 'client':
        client()
    else:
        stand_alone()
    # client()
    # server()
    print('finish')
```

2、 基于 socket()的 TCP 的分布式文件系统

```
import socket
import sys
import os

End=bytes('EOF', encoding='utf-8')

def server():
    HOST = ''    # Symbolic name meaning all available interfaces
    PORT = 8888    # Arbitrary non-privileged port

    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((HOST, PORT))
    s.listen(10)
    print('Socket now listening on port: ' + str(PORT))

    # wait for client connect
    conn, addr = s.accept()
    print('Connected with ' + addr[0] + ':' + str(addr[1]))

    FBASE='filebase_server'

    # now keep talking with the client
    while True:
        # wait to accept command
        cmd = conn.recv(4096)

        if not cmd:
```

```
    print('Disconnected with ' + addr[0] + ':' + str(addr[1]))
    break

cmd_recv = str(cmd, encoding='utf-8').strip()

if cmd_recv == 'upload':
    filename_b = read_till_End(conn)
    conn.sendall(End)

    data_b = read_till_End(conn)

    try:
        with open('%s/%s'%(FBASE, str(filename_b, encoding='utf-8')), 'wb') as f:
            f.write(data_b)
        message = 'upload success'
    except Exception as err:
        message = 'upload failed'

    print(message)
    conn.sendall(bytes(message, encoding='utf-8'))

elif cmd_recv == 'download':
    filename_b = read_till_End(conn)
    filename = '%s/%s' % (FBASE, str(filename_b, encoding='utf-8'))
    if os.path.exists(filename):
        with open(filename, 'rb') as f:
            conn.sendfile(f)
            conn.sendall(End)
        print('download success')
    else:
        conn.sendall(End)
        print('download failed')

elif cmd_recv == 'list':
    files = os.listdir(FBASE)
    file_info = '\n'.join(files)
    conn.sendall(bytes(file_info, encoding='utf-8'))
    conn.sendall(End)
    print(file_info)
else:
    reply = 'command not defined: ' + cmd_recv
    conn.sendall(bytes(reply, encoding='utf-8'))
    print(reply)
```

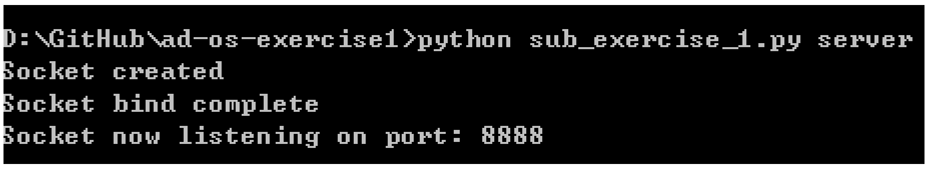
```
        conn.close()
        s.close()

def client():
        host = 'localhost'
        port = 8888

        # create an AF_INET, STREAM socket (TCP)
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

        # establish connect
        remote_ip = socket.gethostbyname(host)
        s.connect((remote_ip, port))
        print('Socket Connected to ' + host + ' on ip ' + remote_ip + ' and port ' + str(port))

        while True:
                try:
                        print('\n 请选择要进行的操作：\n'
                              'upload          upload a file to server file base\n'
                              'download        download a file from server file base\n'
                              'list            list all files available on server\n')
                        cmd = sys.stdin.readline().strip('\n')

                        s.sendall(bytes(cmd, encoding='utf-8'))

                        if cmd == 'exit':
                                break
                        elif cmd == 'upload':
                                print('please specify file path:')
                                filepath = sys.stdin.readline().strip('\n')
                                if not os.path.exists(filepath):
                                        print('file "%s" not exists' % filepath)
                                        continue

                                send_append_End(s,  bytes(os.path.basename(filepath),  encoding='utf-
8'))
                                s.recv(1024)        #  用于阻塞进程
                                with open(filepath, 'rb') as f:
                                        s.sendfile(f)
                                        s.sendall(End)
                                res = s.recv(4096)
                                print(str(res, encoding='utf-8'))
                        elif cmd == 'download':
                                print('please specify file name:')
                filename = sys.stdin.readline().strip('\n')
                send_append_End(s, bytes(filename, encoding='utf-8'))
                data_b = read_till_End(s)
                if len(data_b) == 0:
                    print('file "%s" not exists' % filename)
                else:
                    with open('filebase_client/%s' % filename, 'wb') as f:
                        f.write(data_b)
                    print('download success')

            elif cmd == 'list':
                data_b = read_till_End(s)
                file_info = str(data_b, encoding='utf-8')
                print(file_info)

            else:
                da = s.recv(4096)
                print(str(da, encoding='utf-8'))

        except socket.error:
            # Send failed
            print('Send failed')
            sys.exit()

    s.close()

def send_append_End(s, data_b):
    s.sendall(data_b)
    s.sendall(End)

def read_till_End(s):
    total_data = []
    while True:
        data = s.recv(8192)
        if End in data:
            total_data.append(data[:data.find(End)])
            break
        total_data.append(data)
        if len(total_data) > 1:
            # check if end_of_data was split
            last_pair = total_data[-2] + total_data[-1]
            if End in last_pair:
                total_data[-2] = last_pair[:last_pair.find(End)]
                total_data.pop()
```

```
                    break

        return b''.join(total_data)

    if __name__ == '__main__':
        if len(sys.argv) == 1:
            print('role not specified(server or client)')
            exit()
        if sys.argv[1] == 'server':
            server()
        elif sys.argv[1] == 'client':
            client()
        # client()
        # server()
        print('finish')
```

七、执行结果截图

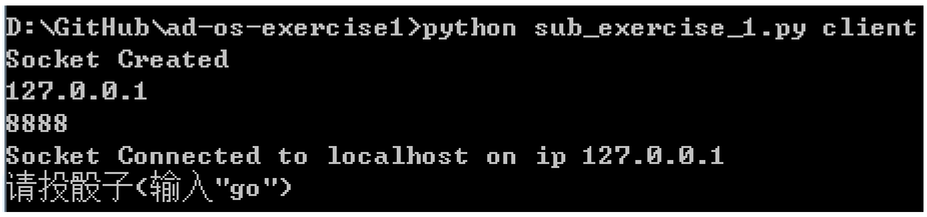
1、 掷骰子

a.启动服务器，开始监听客户端请求：



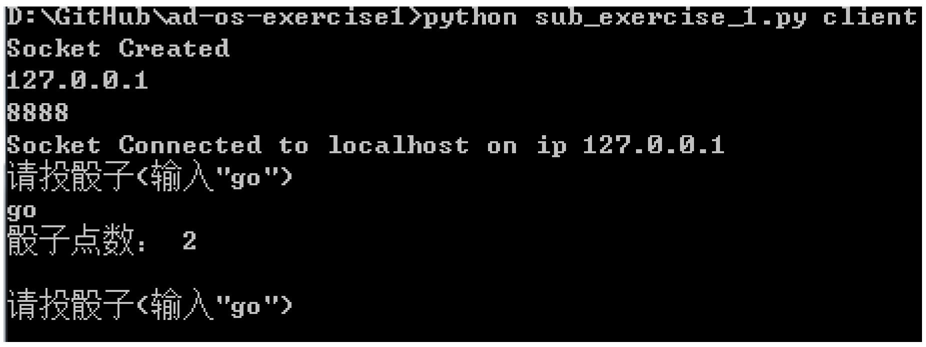
```
D:\GitHub\ad-os-exercise1>python sub_exercise_1.py server
Socket created
Socket bind complete
Socket now listening on port: 8888
```

b.启动客户端，连接服务器：



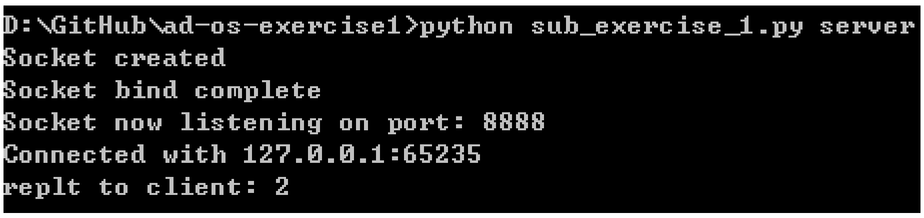
```
D:\GitHub\ad-os-exercise1>python sub_exercise_1.py client
Socket Created
127.0.0.1
8888
Socket Connected to localhost on ip 127.0.0.1
请投骰子(输入"go")
```

c.客户端运行过程



```
D:\GitHub\ad-os-exercise1>python sub_exercise_1.py client
Socket Created
127.0.0.1
8888
Socket Connected to localhost on ip 127.0.0.1
请投骰子(输入"go")
go
骰子点数:  2

请投骰子(输入"go")
```

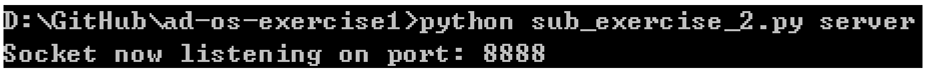
d.服务端运行过程



```
D:\GitHub\ad-os-exercise1>python sub_exercise_1.py server
Socket created
Socket bind complete
Socket now listening on port: 8888
Connected with 127.0.0.1:65235
replt to client: 2
```

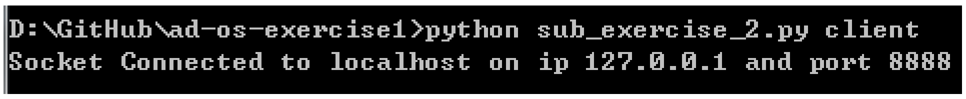
2、 分布式文件系统

a.　启动服务器，开始监听客户端请求：



```
D:\GitHub\ad-os-exercise1>python sub_exercise_2.py server
Socket now listening on port: 8888
```

b.　启动客户端，连接服务器：

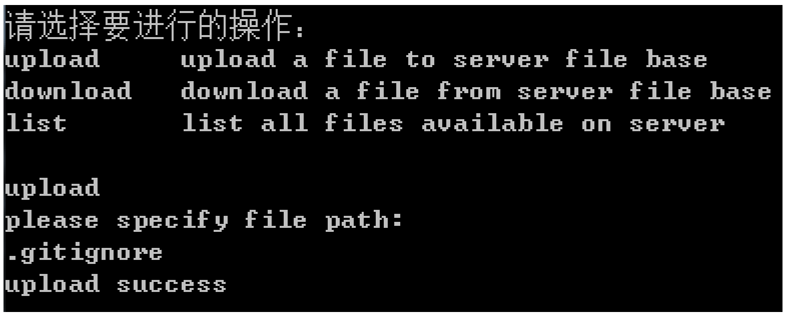


```
D:\GitHub\ad-os-exercise1>python sub_exercise_2.py client
Socket Connected to localhost on ip 127.0.0.1 and port 8888
```

c.运行过程

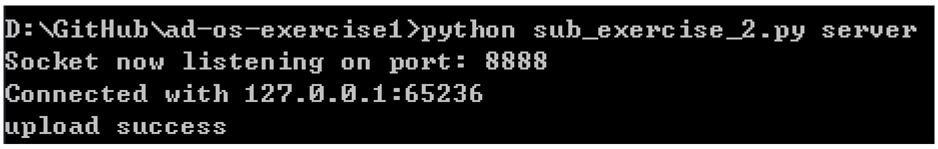
upload：

客户端：



```
请选择要进行的操作：
upload      upload a file to server file base
download    download a file from server file base
list        list all files available on server

upload
please specify file path:
.gitignore
upload success
```

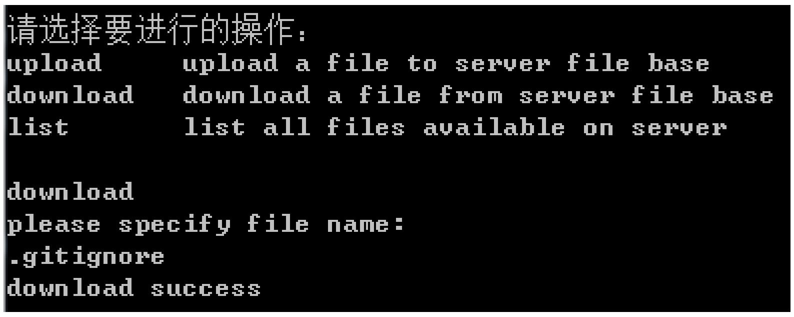
服务端：



```
D:\GitHub\ad-os-exercise1>python sub_exercise_2.py server
Socket now listening on port: 8888
Connected with 127.0.0.1:65236
upload success
```

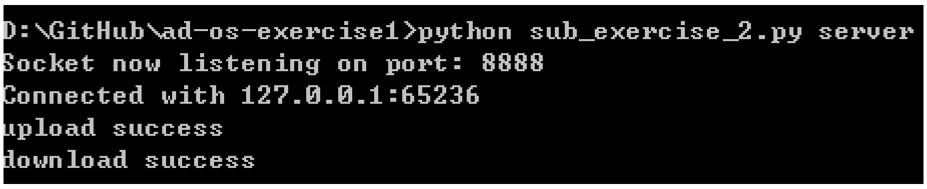
download：

客户端：



```
请选择要进行的操作：
upload      upload a file to server file base
download    download a file from server file base
list        list all files available on server

download
please specify file name:
.gitignore
download success
```

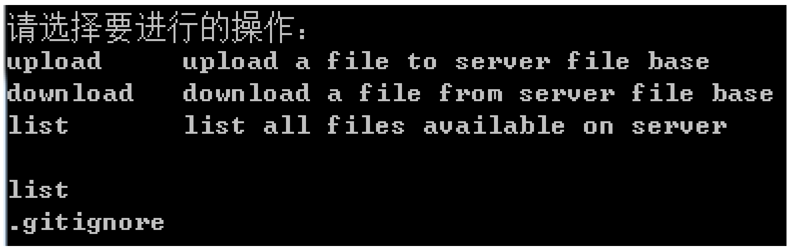
服务端：



```
D:\GitHub\ad-os-exercise1>python sub_exercise_2.py server
Socket now listening on port: 8888
Connected with 127.0.0.1:65236
upload success
download success
```

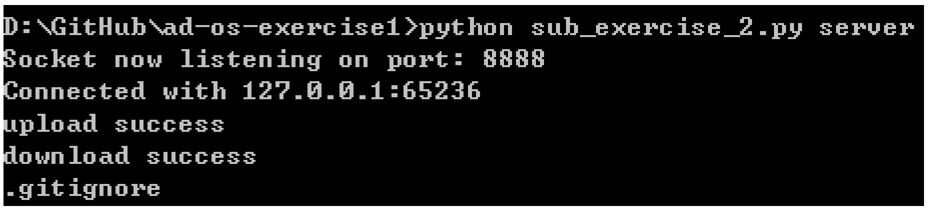
list：

客户端：



```
请选择要进行的操作：
upload     upload a file to server file base
download   download a file from server file base
list       list all files available on server

list
.gitignore
```

服务端：



```
D:\GitHub\ad-os-exercise1>python sub_exercise_2.py server
Socket now listening on port: 8888
Connected with 127.0.0.1:65236
upload success
download success
.gitignore
```

八、使用说明

通过启动 python 程序时指定 client/server 来区分不同的运行方式

九、总结与完善

通过 socket 实现了 client/server 模式的程序，对网络通信、分布式程序、交互流程有了更深的理解。当然，目前服务器端只能连接一个客户端程序，后面可以改成可以利用多线程来相应不同客户端的请求。