当前任务

1. 简述网络七层模型的概念
2. 简述TCP服务和UDP服务区别，三次握手和四次挥手是什么，过程是怎样的？
3. socket套接字编程：
   1. UDP()套接字编程(sbsrc->ssrc)和TCP套接字编程(sblarsc->scsrc)，服务端和客户端使用流程。相关函数的功能参数返回值。
   2. TCP套接字传输特征，网络收发缓冲区是什么？TCP粘包的原因，影响，如何处理？
   3. 套接字属性

type,family,

getsockname(),fileno(),getpeername(),setsocketopt(level,option,value)

getsockoption(level,option)

setblocking(bool) settimeout(sec)

* 1. UDP应用之广播
  2. 本地套接字

1. HTTP协议—超文本传输协议，应用层协议，该协议特点，网页请求过程是怎样的？
   1. HTTP请求

请求行GET / HTTP/1.1

请求头 head

请求体

* 1. HTTP响应

响应行HTTP/1.1 200 OK

响应头

响应内容

* 1. 响应各类响应码 1xx 2xx …

1. IO多路复用
   1. rs, ws, xs = select(rlist, wlist, xlist[, timeout])
   2. poll & epoll
      1. select.poll()->register(fd,event)->unregister(fd)->p.poll() 各自使用，配合字典

使用方式以及能写出具体实例

1. 多进程编程
   1. os.fork() 使用实例，注意事项
   2. 什么是进程，进程和程序的区别，进程有哪些特征，哪些状态，各种状态之间如何转换
   3. 孤儿进程，僵尸进程，如何产生，如何避免
   4. os.wait(), os.waitpid(pid,option)
   5. Process创建进程 start() join() is\_alive(), p.name, p.daemon, p.pid, 实例和使用
2. 群聊聊天室的书写，要求要使用进程并且使用IO多路复用

|  |  |
| --- | --- |
|  | 1. 装饰器的使用方式？ 2. \_\_slots\_\_=(age, name)可以保证该类不能重新再创建其他属性 3. 匿名函数：list(map(lambda x: x\*x, [1,2,3,4,5])) 4. map函数和reduce函数的使用和区别 5. 闭包的使用场景和使用方式 6. 偏函数实际就是把函数部分参数固定下来，比如   import funtools  int2 = functools.partial(int, base=2)  int2(10010)   1. isinstance( 2, int) -> true/false方法的使用，最好不要使用type( ) 2. @property 使用，可以构建属性的装饰器   @property  def width(self):  return self.\_width  @width.setter  def width(self, value):  self.width=value//这里可以做更多的判断来处理   1. python支持多重继承(mixIn) 2. 魔法函数   \_\_repr\_\_ \_\_str\_\_ 一组配合使用  \_\_iter\_\_ \_\_next\_\_一组配合使用，实现可以使用next调用  \_\_getitem\_\_ 实现了就可以使用下标方式来访问   1. slice 是切片类型，判断类型时候可以使用 isinstance(value, slice) 2. <https://www.liaoxuefeng.com/wiki/0014316089557264a6b348958f449949df42a6d3a2e542c000/0014319098638265527beb24f7840aa97de564ccc7f20f6000>后半部分内容，完全动态调用 3. Enum使用   from enum import Enum，unique  @unique #保证唯一性  class Month(Enum):  xxxx   1. 文件和目录操作，使用os模块，里面有   abspath()查看当前目录的绝对路径  os.path.split()可以获取路径和文件名的元组(“/”,”test.txt”)  os.path.splitext(“/text”“.txt”)  os.listdir(‘.’)  os.path.isdir()  os.path.isfile()   1. 序列化对象   import pickle  bytes = pickle.dump(obj)  obj = pickle.load(bytes)  import json  json.dumps()  json.loads()  import json  class Student(object):  def \_\_init\_\_(self, name, age, score):  self.name=name  self.age=age  self.score=score  def \_\_str\_\_(self):  print('xx')  def \_\_repr\_\_(self):  print('xy')  s = Student('jm', 18, 99)  str1 = json.dumps(s, default=lambda obj:obj.\_\_dict\_\_)  print(str1)  def d2s(d):  return Student(d['name'], d['age'], d['score'])  s = json.loads(str1, object\_hook=d2s)  print(s.name) |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |
|  |  |
|  | |