**随堂测试①**

一．单选题（每空3分），共30分。在以下每小题给出的A 、B、C、D四个选项中，只有一个选项正确，请选出正确答案，并将答案填在题中括号内）。

1. Web的底层协议HTTP一般使用端口（B）进行通信。
2. 20
3. 80
4. 21
5. 81
6. 下面那个类可以用于创建直接能访问URL资源的流对象（C），读取网络数据流。
7. BufferedInputStream
8. BufferedReader
9. InputStream
10. DataInputStream
11. 下面不是互联网应用层协议的是（D）
12. FTP
13. HTTP
14. TELNET
15. UDP
16. 下列不属于TCP/IP参考模型的是（C）
17. 网际层
18. 应用层
19. 会话层
20. 主机-网络层
21. 以下IPv6地址不正确的是（C）？
22. AD80::ABAA:0000:00C2:0002
23. 2001:0:0:0:0:0:0:1
24. ::FFFF.192.168.89.9
25. 2000:DB0:2DE::E13
26. 从哪里可以找到描述TCP/IP的具体文档（D）？
27. JDK的JavaDoc文档
28. NIC的官方网站
29. 国际标准化组织（ISO）的官方网站
30. RFC的官方网站
31. 使用Executor线程池编程框架时，定义计算任务的类应该实现（C）接口。
32. Executor
33. Runnable
34. Callable
35. Future
36. 调用join方法的线程是（C）？
37. 主线程
38. 被连接的线程
39. 连接线程
40. 副线程
41. 以下关于线程相关内容说法正确的是（B）？
42. 同步块越大越好
43. 轮询和回调都是解决静态条件的有效方法
44. 对于Thread的派生子类，应该在编程时覆盖Thread的所有方法。
45. 线程优先级中，1表示优先级最高。
46. 有一种协议规定：如果客户端发送一行字符串“date”,服务器端就返回当前日期信息，如果客户端发送一行字符串“exit”，服务器端就结束与客户端的通信。这种协议应该是哪一层的协议（A）？
47. 应用层
48. 网际层
49. 主机-网络层
50. 传输层

二．判断题 （每小题2分，共10分，请给正确的打“√”，错误的打“×”，并填写题中括号内）。

1. 假定一个进程已经占用TCP的80端口，它就不可以占用UDP的80端口（ F ）
2. DataOutputStream类提供了8字节的写入方法，并且以big-endian格式写入 （ T ）
3. SMTP是发送电子邮件的协议（ T ）
4. 在关闭流之前不需要进行刷新操作（ F ）
5. 避免死锁可以利用同步方法（ F ）

三．综合题（每小题20分，共60分）

1. 请结合实例说明什么是网络地址转换（NAT），其主要作用是什么？

网络地址转换（Network Address Translation，简称NAT）是一种常用的网络技术，用于在互联网与局域网之间进行IP地址转换。

NAT的主要作用是解决IP地址不足的问题。在IPv4协议下，IP地址资源有限，不足以满足所有设备的需求。NAT通过将内部局域网中的私有IP地址转换成公有IP地址，实现了内部网络与外部网络之间的通信。

以下是一个具体的示例：

假设有一个公司内部的局域网，其中有多台计算机需要连接到互联网。这个局域网使用私有IP地址范围（如192.168.0.0/24），而公司只有一个公有IP地址供整个局域网使用。

当局域网中的一台计算机发送请求到互联网上的服务器时，NAT会将源IP地址（192.168.0.2）替换成公有IP地址（例如203.0.113.1），并在NAT设备上记录这个映射关系。服务器接收到请求后，将响应发送给公有IP地址，再由NAT设备根据之前记录的映射关系将响应转发给对应的局域网内计算机（192.168.0.2）。

这样，通过NAT技术，局域网内的多台设备可以共享一个公有IP地址与互联网通信，大大节约了IP地址资源的使用。

总结起来，NAT的主要作用是将内部私有IP地址转换为公有IP地址，实现内部网络与外部网络之间的通信，并解决IP地址不足的问题。

1. 在程序中直接将读入的网络数据流转换为等长字节数组存在安全问题，因为在网络获取数据不流畅，数据流的传送会断断续续，不能保证一次能读取全部数据。特别是在读取大容量网络数据时问题更严重。假设有以下场景，应用程序servelet当前收到http（协议）请求流，请设计程序怎么保证当前程序接收到所有请求流数据？请写出主要代码和思路。

已知http请求流的获得指令：

ServletInputStream inStream = request.getInputStream(); //取HTTP请求流

int size = request.getContentLength(); //取HTTP请求流长度

参考思路：

在读取数据时检测实际读到的长度，如果没有读完已知长度的数据就应该再次读取，以此循环检测，直到实际读取的长度累加与已知的长度相等。

1. ServletInputStream inStream = request.getInputStream(); //取HTTP请求流
2. **int** size = request.getContentLength(); //取HTTP请求流长度
3. byte[] buffer = **new** byte[size]; //用于缓存每次读取的数据
4. byte[] in\_b = **new** byte[size]; //用于存放结果的数组
5. **int** count = 0;
6. **int** rbyte = 0;
7. **while** (count < size) { //循环读取
8. rbyte = inStream.read(buffer); //每次实际读取长度存于rbyte中
9. **for**(**int** i=0;i;)
10. in\_b[count + i] = buffer[i];
11. }
12. count += rbyte;
13. }
14. Java实现线程池有哪些类？试画出相应实现程序的类结构图，并简要说明他们之间的关系。

在Java中，实现线程池的主要类有以下几个：

Executor 接口：定义了执行任务的基本方法。所有的线程池实现类都实现了这个接口。

ExecutorService 接口：继承自Executor接口，提供了更丰富的任务执行控制方法，例如提交任务、关闭线程池等。

ThreadPoolExecutor 类：是Java内置的线程池实现类，实现了ExecutorService接口。它可以通过构造函数自定义线程池的各个参数，如核心线程数、最大线程数、线程空闲时间等。

ScheduledExecutorService 接口：是ExecutorService接口的子接口，可以定时执行任务或周期性执行任务。

下面是它们之间的类结构图（简化版）：

+----------------+

| Executor |

+----------------+

^

|

+----------------+

| ExecutorService|

+--^-------------+

|

+----------------+-------------+

| ThreadPoolExecutor |

+----------------+-------------+

|

|

+--------------|--------------+

| ScheduledExecutorService|

+---------------------------+

在这个类结构中，Executor接口是基础，定义了线程池的最基本的执行方法。ExecutorService接口扩展了Executor接口，提供了更多的任务执行控制方法，例如提交任务、关闭线程池等。

ThreadPoolExecutor是ExecutorService接口的实现类，是Java内置的线程池实现。它可以通过构造函数自定义线程池的各种参数。

ScheduledExecutorService接口继承自ExecutorService接口，提供了定时执行任务或周期性执行任务的功能。

这些类之间的关系是：ThreadPoolExecutor实现了ExecutorService接口，而ExecutorService接口继承自Executor接口。ScheduledExecutorService接口是ExecutorService接口的子接口，提供了额外的定时任务功能。