进程：具有一定独立功能的程序关于某个数据集合上的一次运行活动，是系统进行资源分配和调度的一个独立单元。

多线程编程实现①：创建线程对象，执行start()方法，需要定义Thread子类，并实现run()方法。

1. Thread t = **new** DigestThread();
2. t.start();
3. **class** DigestThread **extends** Thread {
4. **public** **void** run() {
5. }
6. }

多线程编程实现②：实现Runnable接口。

1. **class** DigestRunnable **implements** Runnable{
2. **public** **void** run() {
3. }
4. }
5. Thread t = **new** Thread(**new** DigestRunnable());
6. t.start();

同步带来的问题：①程序性能下降，甚至严重下降；②增加了死锁的可能性；③可能不能有效保护数据。

死锁：两个线程需要独占访问同样的资源集。

防止死锁：①避免不必要的同步；②同步块尽可能的小；③尽可能不一次同步多个对象。

线程优先级：0-10，默认为5，最小为1.

线程池优点：①降低线程创建和释放的开销；②避免运行过多线程超出计算机处理能力。

1. ExecutorService pool = Executors.newFixedThreadPool(ThREAD\_COUNT);
2. for(int i=0;i<files.length;i++){
3. if(!files[i].isDirectory()){
4. Runnable task=new GZipRunnable(files[i]);
5. Pool.submit(task);
6. }
7. }

字符串构造URL：

1. try {
2. URL u = new URL("http://www.szu.edu.cn/");
3. }
4. catch (MalformedURLException ex) {
5. System.err.println(ex);
6. }

分块构造URL：

1. try {
2. URL u = new URL("http", "www.eff.org", "/blueribbon.html#intro");
3. }
4. catch (MalformedURLException ex) {
5. throw new RuntimeException("shouldn't happen; all VMs recognize http");
6. }

构造相对URL：

1. try {
2. URL u1 = new URL("http://www.szu.edu.cn/a/b.html");
3. URL u2 = new URL(u1, "mailinglists.html");
4. }
5. catch (MalformedURLException ex) {
6. System.err.println(ex);
7. }

从URL中获取数据：openStream

1. try {
2. URL u = new URL("http://www.szu.edu.cn");
3. InputStream in = u.openStream();
4. int c;
5. while ((c = in.read()) != -1)
6. System.out.write(c);
7. in.close();
8. }catch (IOException ex) {
9. System.err.println(ex);
10. }

从URL中获取数据：openConnection

1. try {
2. URL u = new URL("http://www.szu.edu.cn/");
3. try {
4. URLConnection uc = u.openConnection();
5. InputStream in = uc.getInputStream();
6. // read from the connection...
7. } catch (IOException ex) {
8. System.err.println(ex);
9. }
10. } catch (MalformedURLException ex) {
11. System.err.println(ex);
12. }

TCP：面向连接、面向字节流、支持全双工、支持并发连接、提供确认重传与拥塞控制的可靠传输层协议。

保证数据到达、保证数据顺序、速度慢、用途：HTTP、FTP

UDP：无连接、不可靠、面向报文的协议。

不保证一定到达、不保证到达顺序、速度块、用途：DNS、NFS、TFTP

日志模块：

1. public class Logger {
2. private final String log;
3. private final String filePath;
4. public Logger(String IP, Date date,String filePath){
5. log=IP+" login at "+date.toString()+'\n';
6. this.filePath=filePath;
7. keep();
8. }
9. private synchronized void keep(){
10. try {
11. BufferedWriter bufferedWriter=new BufferedWriter(new FileWriter(filePath,true));
12. bufferedWriter.write(log);
13. bufferedWriter.close();
14. } catch (IOException e) {
15. e.printStackTrace();
16. }
17. }
18. }