Java NIO  
>>NIO

1. 基础IO操作中的所有操作都是直接以流 的方式完成的，而在NIO中的所有操作都要使用到缓冲区处理，且所有的读写操作都是通过缓冲区完成的。
2. 缓存区类：
   1. ByteBuffer
   2. CharBuffer
   3. ShortBuffer
   4. IntBuffer
   5. LongBuffer
   6. FloatBuffer
   7. DoubleBuffer
3. 各个缓冲类提供的常用方法
   1. allocate(int capacity)：创建一个的NIO类，容量capacity；当开辟的空间小，但是写入的数据大的时候，会BufferOverflowException
   2. allocateDirect(capacity):开辟直接缓冲区，只有ByteBuffer有，将缓冲区建立在物理内存中，避免中间重复的copy操作。
   3. get()：
   4. get(index)
   5. put(T x)：向缓冲区中加入数据
   6. put(T[] x)：向缓冲区加入数组
   7. put(T[] x,offset,length)：
   8. slice():将postion和limit划分为一个子缓冲区，然后可以进行局部操作子缓冲区的内容。
   9. asReadOnlyBuffer():返回一个只读的NIO对象
   10. flip():重设缓冲区，positon指向0，limit指向数组已有数据的下一个位置
   11. hasRemaining():判断缓冲区是否还要数据
4. 通道：既可以完成输入也可以完成输出工作,FileChannel,可以通过FileOutPutStream.getChannel()获得
   1. close()
   2. isOpen()
   3. read(ByteBuffer dst)
   4. write(ByteBuffer src)
   5. map(FileChannel,MapMode,position,size)
5. 内存映射MappedByteBuffer读取文件内容最快
   1. 读取文件的四种方式：RandomAccessFile，FileInputStream，BufferReader，MapperByteBuffer
   2. FileChannel的3中内存映射模型，
      1. MapMode.READ\_ONLY:只读模式
      2. MapMode.READ\_WRITE:读取/写入映射模式
      3. MapMOde.PRIVATE:专用映射模式
   3. MappedByteBuffer mbb = fileChannel.map(FileMap.MapMode.READ\_ONLY);
6. 文件锁FileLock
   1. 共享锁：允许多个线程进行文件的读取操作
   2. 独享锁：只允许一个线程进行文件的读写操作