예외

예외가 발생시 하단 내용 실행 안됨을 유의

printStackTrace()
getMessage()
getStackTrace()

```
try{
}catch(Exception e){
e.printStackTrace();
}finally{}
```

throws Exception은 예외에 대한 책임을 호출한 쪽으로 넘김

자료구조

리스트 형식

큐, 스택 기능

맵 기능

트리구조

Generic이란? <>

리스트 : 순서를 가지고 있음, 중복 데이터 가능
세트 : 순서 없음, 중복 불가능
맵 : 키와 값을 가짐, 키를 이용하여 검색

각 구조는 인터페이스로 정의

Element? 자료구조 안으로 들어가는 데이터를 의미, 자바에서는 레퍼런스

리스트

```
ArrayList<String> list = new ArrayList<String>();
list.add("AAAA");
list.add("BBBB");
list.add("CCCC");
list.add("DDDD");
System.out.println(list);
```

```
[AAAA, BBBB, CCCC, DDDD]
```

```
ArrayList<String> list = new ArrayList<String>();
list.add("AAAA");
list.add("BBBB");
list.add("CCCC");
list.add("DDDD");
list.set(1,"EEEE");
System.out.println(list);
```

```
[AAAA, EEEE, CCCC, DDDD]
```

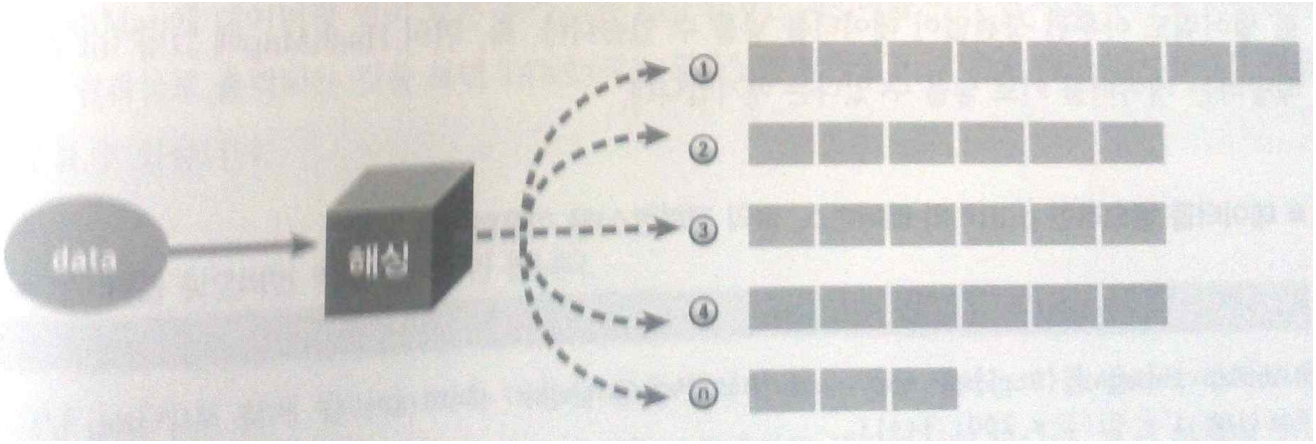
리스트

```
ArrayList<String> list = new ArrayList<String>();
list.add("AAAA");
list.add("BBBB");
list.add("CCCC");
list.add("DDDD");
String str = list.get(1);
System.out.println(list);
```

```
list: [AAAA, BBBB, CCCC, DDDD]
str: BBBB
```

```
public class SampleObj {
    private String name;
    public SampleObj(String name){
        this.name = name;
    }
    public String toString(){
        return name;
    }
    public boolean equals(Object obj){
        String currentObj  = this.toString();
        String otherObj = obj.toString();
        return currentObj.equals(otherObj);
    }
}
```

HashMap




```
for(int i = 0; i < 10; i++){
      map.put(i , "AAAA"+i);
}

map.put(0,"BBBB");
map.put(0,"CCCC");
map.put(0,"DDDD");

System.out.println(map);
```

```
{0=DDDD, 1=AAAA1, 2=AAAA2, 3=AAAA3, 4=AAAA4, 5=AAAA5, 6=AAAA6, 7=AAAA7, 8=AAAA8,
 9=AAAA9}
```

Set

```
HashSet<String> set = new HashSet<String>();

set.add("A");
set.add("B");
set.add("C");
set.add("D");
set.add("E");
set.add(new String("A"));
System.out.println(set);
```

```
[D, E, A, B, C]
```

예제

```
public class Player {

    private String name;
    private String nation;

    public Player(String name){
        this(name, "");
    }
    public Player(String name, String nation){
        this.name = name;
        this.nation = nation;
    }
    public String toString(){
        return name+":"+nation;
    }
    public boolean equals(Object obj){
        String compareValue = obj.toString();
        String thisValue = toString();
        System.out.println("equals" );
        return thisValue.equals(compareValue);
    }
    public int hashCode(){
        return toString().hashCode();
    }
}
```

정렬

자바와 C와의 정렬은 방법이 다르다

```
public class PlayerVO implements Comparable<PlayerVO>{
	private String name;
	private String position;
	private int regYear;

	public PlayerVO(String name, String position, int regYear) {
		this.name = name;
		this.position = position;
		this.regYear = regYear;
	}
	public String toString(){
		return name+":"+position+":"+regYear;
	}

	@Override
	public int compareTo(PlayerVO otherPlayer) {
		System.out.println("정렬시도");
		return this.name.compareTo(otherPlayer.name);
	}
}
```

정렬

```
import java.util.ArrayList;
import java.util.Collections;

public class PlayerSortTest {
      public static void main(String[] args) {
            ArrayList<PlayerVO> list = new ArrayList<PlayerVO>();

            list.add(new PlayerVO("홍길동","투수",1999));
            list.add(new PlayerVO("임꺽정","포수",2005));
            list.add(new PlayerVO("강감찬","1루수",2003));
            list.add(new PlayerVO("을지문덕","2루수",2010));
            System.out.println(list);
            Collections.sort(list);
            System.out.println(list);
      }
}
```

파일복사 프로그램

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.io.OutputStream;

public class FileCopy1 {
    public static void main(String[] args) throws Exception{
        //읽어들일 수 있는 스트림과 출력 가능한 스트림을 준비한다.
        InputStream in = null;
        OutputStream out = null;

        try{
            in = new FileInputStream("C:\\zzz\\aaa.jpg");
            out = new FileOutputStream("copy.jpg");
```

파일복사 프로그램

```
            while(true){
                int data = in.read();
                if(data == -1){
                    break;
                }
                out.write(data);
            }//end while

        }catch(Exception e){
            e.printStackTrace();
        }finally{
            if(in != null){
                try { in.close();}catch(Exception e){}
            }
            if(out != null){
                try { out.close();}catch(Exception e){}
            }
        }//end finally
    }//end main
}
```

인터넷에 대신 연결?

URL url = new URL(http://www.daum.net);
in = url.openStream();

객체를 읽거나 쓸 수 있는 ObjectInputStream, ObjectOutputStream

반드시 Serializable 인터페이스를 구현해야 함

RandomAccessFile 객체
- 읽는 작업, 쓰는 작업이 하나로 가능
- 위치 조정이 마음대로

네트워킹

TCP? UDP?

소켓?

패킷?

프로토콜?

ServerSoket? (리스너)

네트워킹

```
import java.net.ServerSocket;
import java.net.Socket;
// 예외 처리는 생략(반드시 try ~ catch ~ finally로 처리해야 합니다.)
public class ServerSocketTest {
      public static void main(String[] args) throws Exception{
            ServerSocket serverSocket = new ServerSocket(8111);
            System.out.println("클라이언트 연결 대기 중");
            Socket socket = serverSocket.accept();
            System.out.println("클라이언트 연결: " + socket);
            socket.close();
            serverSocket.close();
      }
}
```

```
클라이언트 연결 대기 중
```

스레드

1. 동시에 진행되었으면 하는 작업 선정

2. 클래스 뒤에 extends Thread 또는 implements Runnable을 사용

3. public void run() 메소드를 오버라이드 함

4. 원하는 만큼의 스레드를 만들어서 스타트
 - start()를 호출하면 자동으로 run()메소드가 실행

* 먼저 클래스를 만들고 스래드가 필요한 부분만 수정해서 작성함

경마 게임? (각 말은 랜덤하게 숫자 발생시켜서 더함)

Thread.sleep()이 없으면 한쪽 실행 후 다른쪽 실행 (씨피유 속도 때문에)

synchronized 키워드로 스래드의 독점 관리 가능 (단, 무한 Lock문제 발생 가능성 있음)

스레드

```
public class SumEx implements Runnable{
	@Override

	public void run() {
		try {
			doJob();
		} catch (Exception e) {
			e.printStackTrace();
		}
	}

	public void doJob()throws Exception{
		int sum  = 0;
		for(int i = 1; i <= 100; i++){
			sum = sum + i;
			Thread.sleep(100);
			System.out.println(sum);
		}
	}

	public static void main(String[] args) throws Exception{
		SumEx ex = new SumEx();
		ex.doJob();
	}
}
```

스레드

```
public static void main(String[] args) throws Exception{
        SumEx ex = new SumEx();
        //ex.doJob();
        Thread t0 = new Thread(ex);
        Thread t1 = new Thread(ex);
        t0.start();
        t1.start();
        System.out.println("-------------------------");
}
```

```
-------------------------
1
3
1
6
```

ANT

```
<?xml version="1.0"?>
<project name="mytest" default="compile">
	<property file="build.properties"/>
	<target name="compile" depends="init">
		<javac srcdir="." destdir="${main.classpath}">
			<classpath refid="classpath"/>
		</javac>
		<echo>compilation is completed!</echo>
	</target>

	<target name="init">
		<mkdir dir="classes"/>
	</target>

	<path id="classpath">
	    <fileset dir="${jar.home}">
	        <include name="tools.jar"/>
	    </fileset>
	    <pathelement location="${main.classpath}"/>
	</path>

</project>
```

ANT

property – 속성 지정

mkdir – 새 디렉토리 생성

copy – 파일, 디렉토리 복사

javac – 컴파일

jar – jar파일 생성

javadoc – javadoc 생성

delete – 파일, 디렉토리 삭제