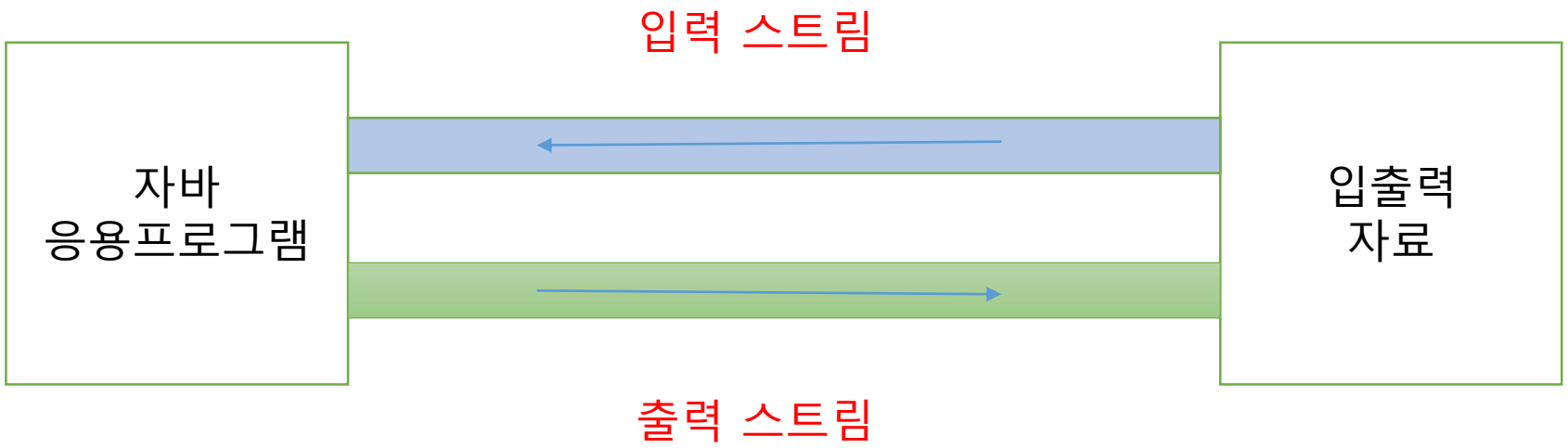
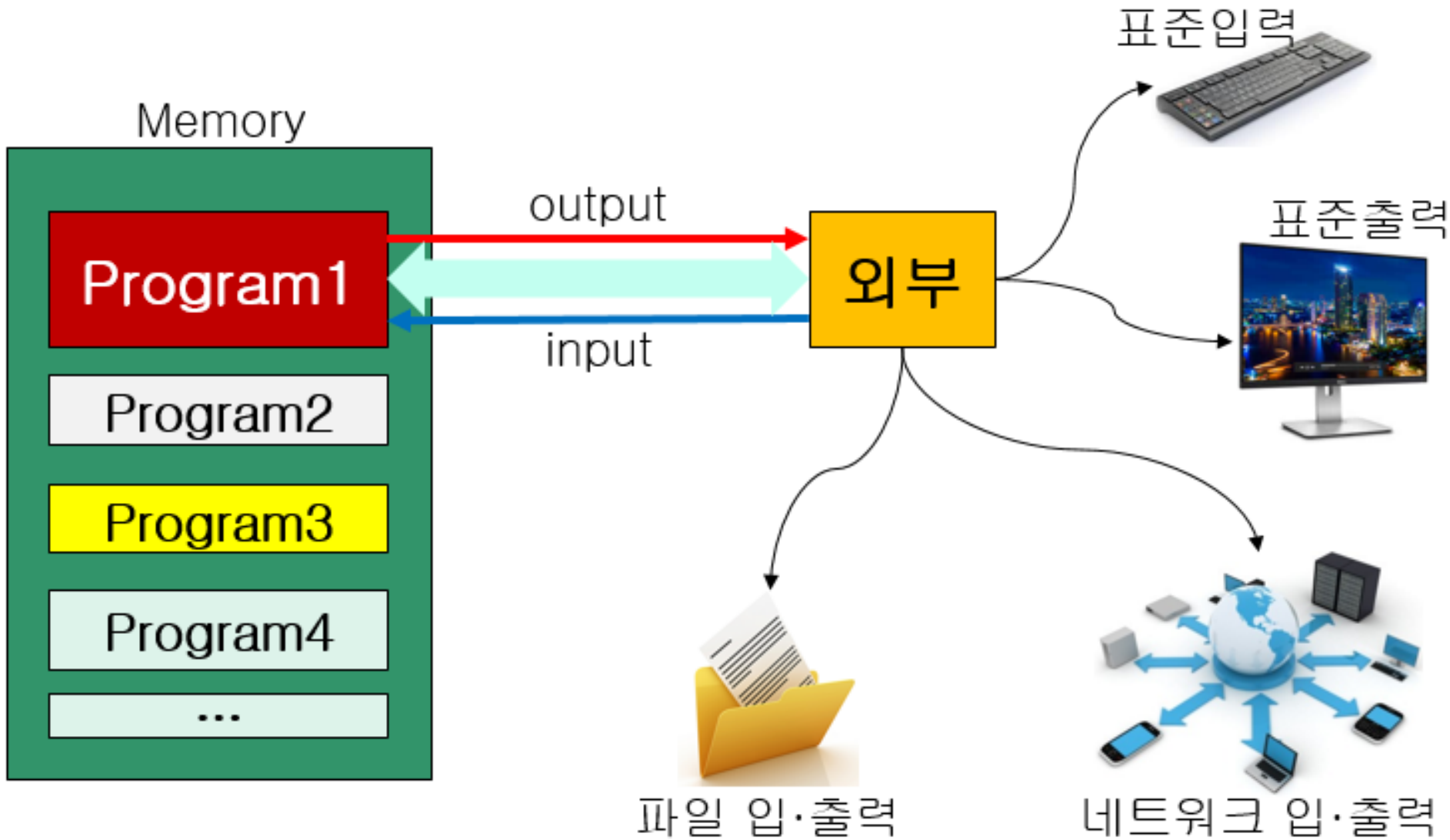
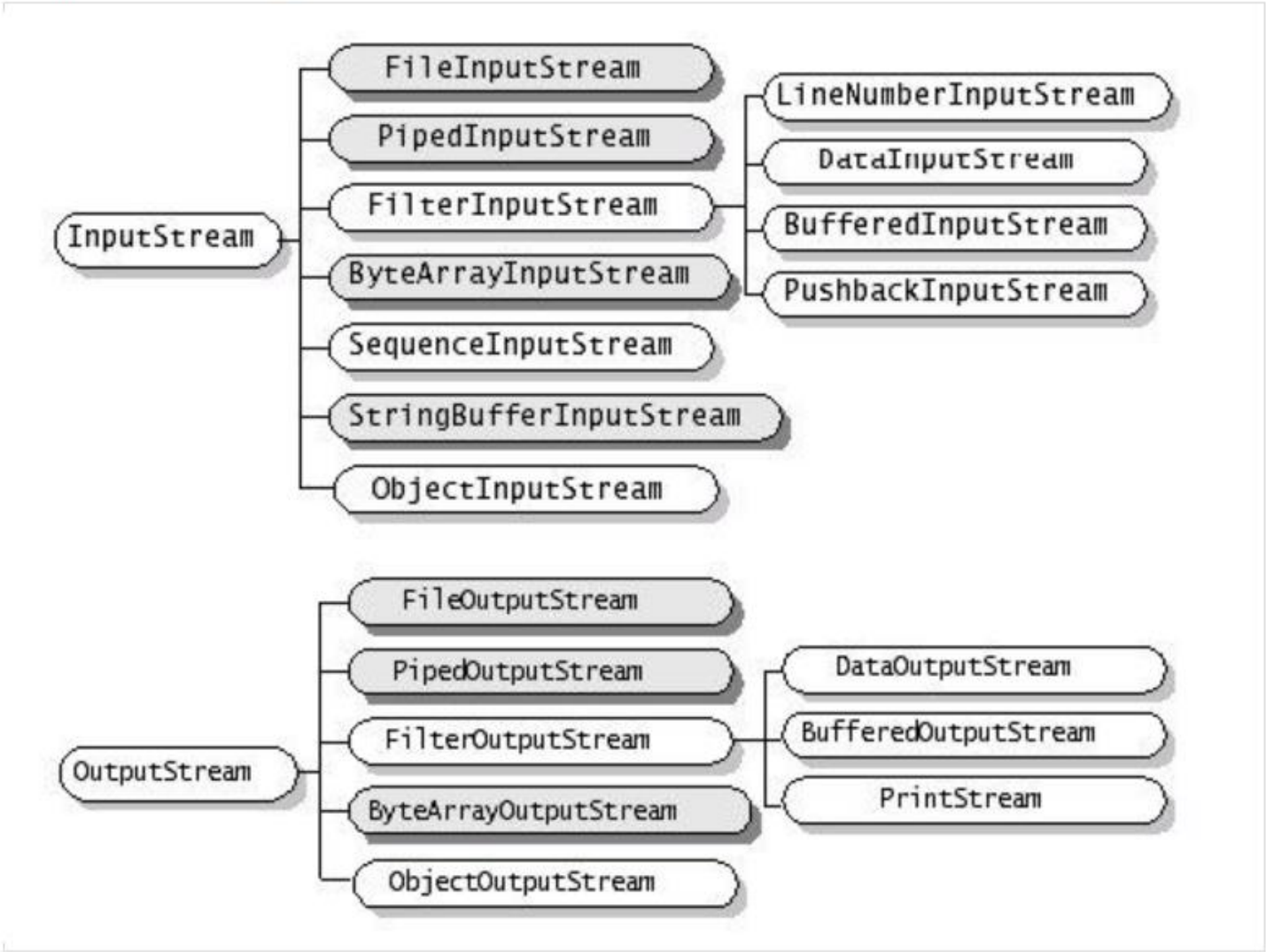
자바의 입출력

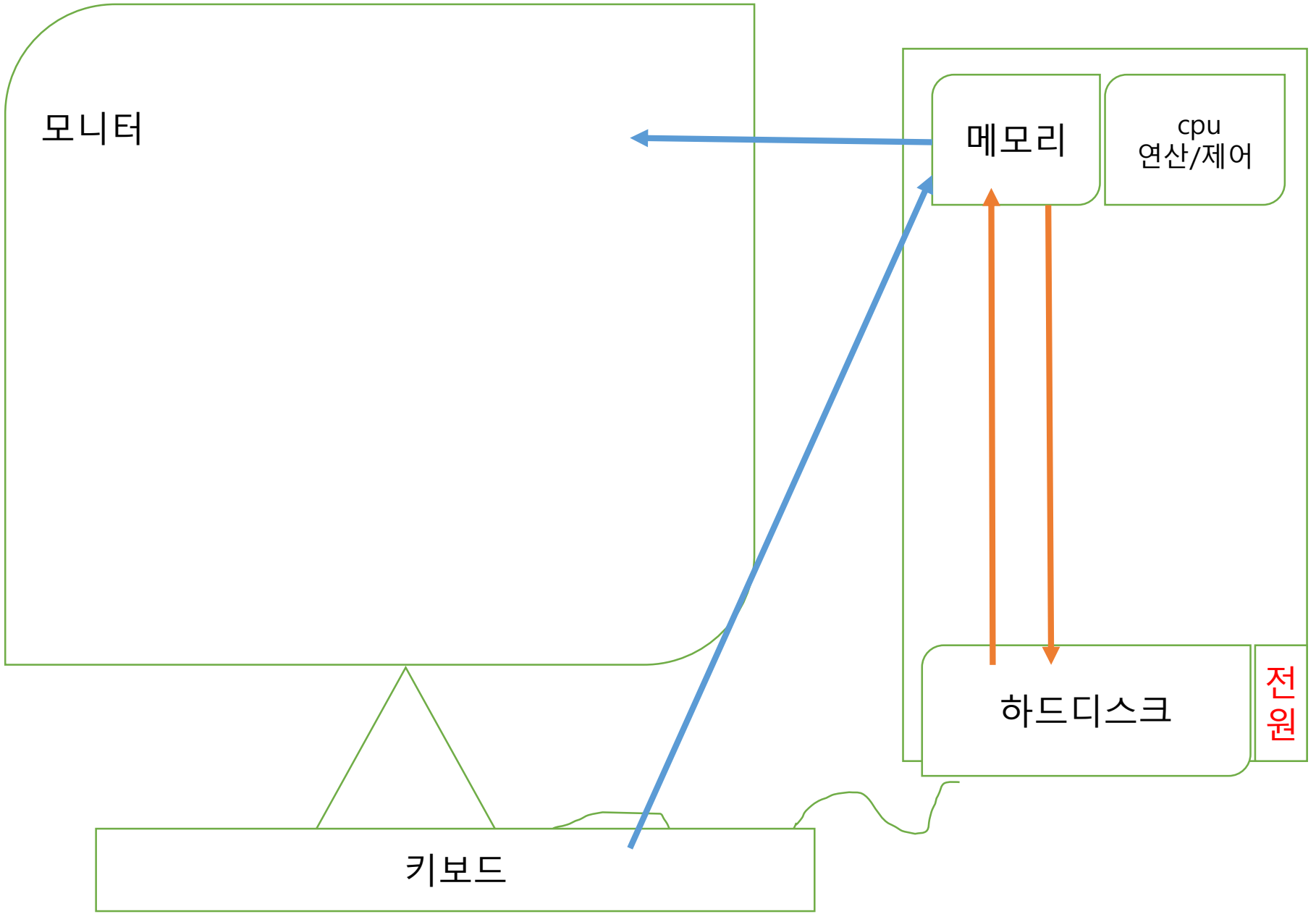


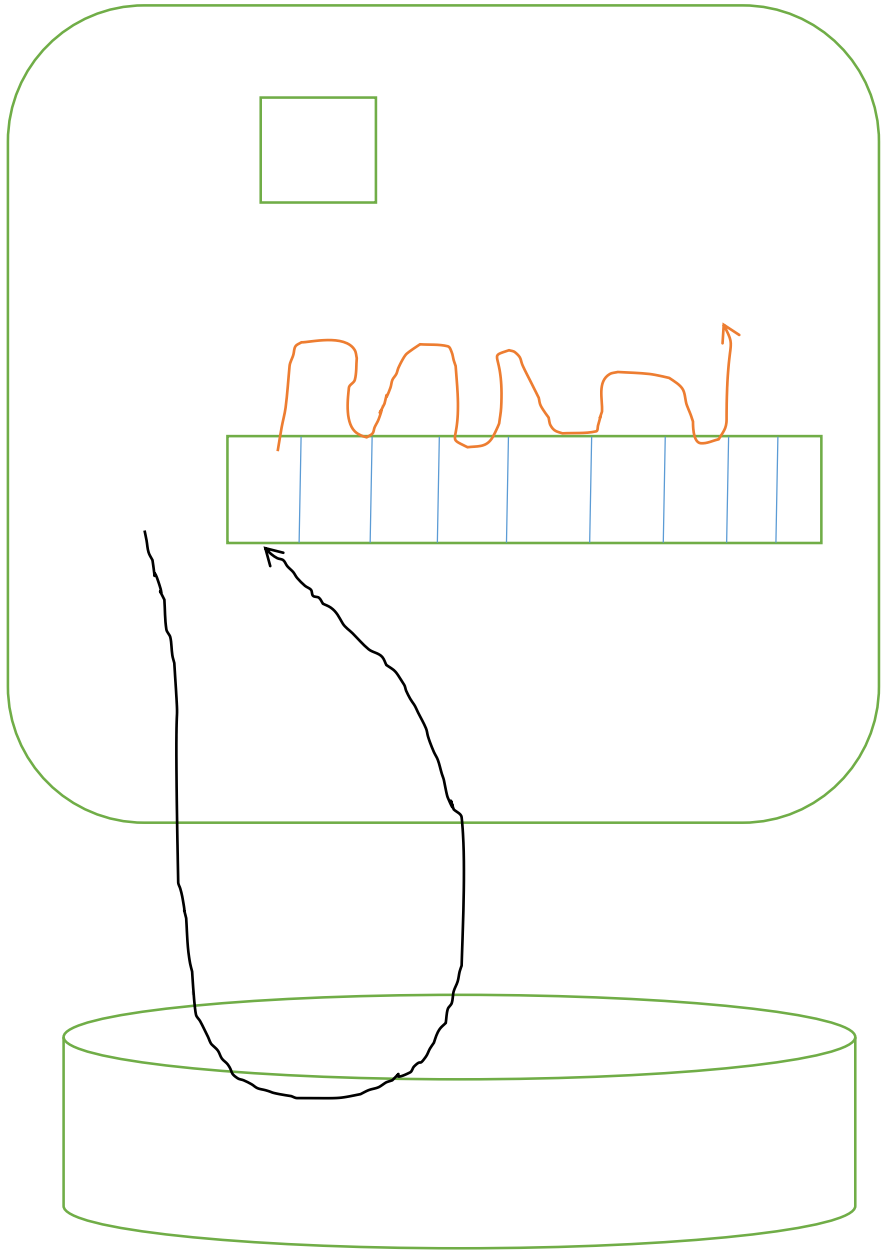

Memory
Program1
Program2
Program3
Program4
...
output
input
외부
표준입력
표준출력
파일 입·출력
네트워크 입·출력

바이트 스트림 (저수준 입출력)

- ✓ 1 바이트(8 Bit)를 읽고 쓰기 위한 스트림
- ✓ 2진 데이터 입출력 가능
- ✓ InputStream, OutputStream 클래스(추상 클래스)와 그 하위 클래스를 이용




모니터
메모리
cpu
연산/제어
하드디스크
전원
키보드



바이트 단위 스트림

자바 스트림은 바이트 단위로 자료의 입출력이 이루어짐
그림, 동영상, 음악파일 등 대부분 파일은 바이트 단위로 읽거나 쓰면 됩니다.
그러나 한글같은 경우는 한 문자, 즉 2바이트 이기때문에 깨져서 나타납니다.
따라서 입출력 중 가장 많이 사용하는 자료인 문자를 위해 문자스트림을 별도로 제공합니다.

문자 단위 스트림

- **기반스트림과 보조스트림**

어떤 스트림이 **자료를 직접 읽거나 쓰는 기능을** 제공하는가 아니면 직접 자료를 읽거나 쓰는 기능 없이 **다른 스트림에 부가기능을 제공하는가에 따라** 기반 스트림과 보조 스트림으로 구분할 수 있습니다.
기반스트림은 입출력 대상에 직접연결되어 생성되는 스트림입니다. 반면 보조 스트림은 읽고 쓰는 기능은 없습니다. 따라서 항상 다른 스트림을 포함하여 생성됩니다.

표준 입출력

System클래스

static PrintStream	out	표준 출력 스트립
static InputStream	in	표준 입력 스트림
static OutputStream	err	표준 오류 출력 스트림

프로그램실행하면 자동으로 생성해줌

```
public class SystemInTest {
	public static void main(String[] args) {
		int input;
		try {
			input= System.in.read();
			System.out.println(input);
			System.out.println((char)input);

		} catch (IOException e) {
			e.printStackTrace();
		}
	}

}
```

```
public class SystemInTest2 {

public static void main(String[] args) {
	System.out.println("알파벳 여러 개를 쓰고 enter를 누르세요");
	int input;
	try {
		while(  ( input =System.in.read()) != '₩n')  {
			System.out.print( (char) input);
		}
	} catch (IOException e) {

		e.printStackTrace();
	}

	}

}
```

BufferedReader in = new BufferedReader(new InputStreamReader (System.in))

1. System.in

BufferedReader in = new BufferedReader(new InputStreamReader (System.in));
키보드와 같은 사용자로부터 입력을 받는 표준입력스트림 객체를 나타냅니다. 그래서, System.in의 리턴값은 InputStream입니다.※
InputStream : 입력스트림으로 부터,데이터를 byte단위로 읽어오는 byte stream입니다.

2. InputStreamReader
BufferedReader in = new BufferedReader(new InputStreamReader (System.in));

InputStreamReader는 인자로, InputStream을 취해서,Reader 스트림형태로 변환합니다. 변환시, 문자열인코딩을 줄수도있습니다. 그럼, Reader스트림이란, InputStream과는 달리, 입력스트림에서, 데이터를 character단위로,처리합니다.

3. BufferedReader
BufferedReader in = new BufferedReader(new InputStreamReader (System.in));

BufferedReader는 인자로 취한 Reader 스트림에 버퍼링기능을 추가한 입력스트림 클래스 입니다.
버퍼를 둠으로써, 파일,네트워크와 같은 물리적인 장치에서 데이터를 사용자가 요청할때마다 매번 읽어오는것보단, 일정량사이즈로 한번에 읽어온후, 버퍼에 보관합니다. 그리고, 사용자가 요구시,버퍼에서 읽어오게됩니다. 결국, 속도를 향상시키고, 시간의 부하를 줄일수 있게 됩니다.
그리고, 사용자가 사용하기 편리한, readLine()과 같이 한줄씩읽어오는 메소드를 제공합니다.

바이트 단위 스트림

```
public class FileInputStreamTest {

	public static void main(String[] args) {

		try {
			FileInputStream fis = new FileInputStream("a.txt");
			int input;

			while ( ( input=fis.read()) != -1) {
				System.out.print((char)input);
			}
		    System.out.println("end");

		}catch(IOException e) {
			e.printStackTrace();
		}

	}

}
```

```
public class FileInputStreamTest2 {

	public static void main(String[] args) {
		try {
			FileInputStream fis = new FileInputStream("input.txt");
			byte[] bs = new byte[10];

			int input;
			while( (input = fis.read(bs)) != -1) {

				for(byte b :bs) {
				System.out.print( (char)b);
			}

				System.out.println( input +"바이트 읽음");
			}

			}catch(Exception e) {
				e.printStactTrace();
			}
	}

}
```

```
class FileOutputStreamTest3 {

	public static void main(String[] args) {



	try {
		FileOutputStream fos = new FileOutputStream("output2.txt");
		byte[] bs = new byte[25];
		byte data =65;
		for( int i=0 ; i< bs.length; i++) {
			bs[i]=data;
			data++;
		}
		fos.write(bs);
	}catch(IOException e) {
		e.printStackTrace();
	}


	}

}
```

FileReader

문자단위스트림

```
public class FileInReaderTest {

	public static void main(String[] args) {

		try {
			FileReader fis = new FileReader("a.txt");
			int input;

			while (  ( input=fis.read()) != -1) {
				System.out.print((char)input);
			}
			System.out.print("end");

		}catch(IOException e) {
				e.printStackTrace();
		}

	}

}
```

```
public class FileWriterTest {

	public static void main(String[] args) {

		try {
			char  buf[] = { 'B', 'A', 'C','F'};
			FileWriter fw = new FileWriter("writer.txt");
			fw.write('A');
			fw.write("안녕하세요");
			fw.write( buf);
			fw.write( buf,1,2);
			fw.flush();

		}catch(IOException e) {
				e.printStackTrace();
		}

	}

}
```

보조스트림 사용

```
public class BufferedReaderTest {

    public static void main(String[] args) {

        BufferedReader bs = new BufferedReader(new InputStreamReader(System.in));
        String input;
        try {
            while( true ) {
             input = bs.readLine();
             if( input.equals("stop"))
                break;
             System.out.println( input);
             }
        } catch (IOException e) {
                e.printStackTrace();
        }
    }
}
```

보조스트림

기반스트림