실력 상승! 연습 문제

Q1 파일의 경로를 저장하기 위해 다음과 같이 path1 문자열을 작성했다. 이를 File.separator 파일 구분자를 사용해 동일한 경로를 나타내는 문자열로 표기하시오.

```
String path1 = "D:\\abc\\bcd\\cde.txt";
String path2 = "D:"+File.separator+"abc"+File.separator+"bcd"+File.separator+"cde.txt";

System.out.println(path1);
System.out.println(path2);
```

separator modification

실행 결과

```
D:\abc\bcd\cde.txt
D:\abc\bcd\cde.txt
```

Q2 현재의 작업 위치는 E:/work 폴더다. 이때 다음 코드의 실행 결과를 쓰시오.

```
File file = new File("mydata/result.txt");
System.out.println(file.getAbsolutePath());
```

E:\work\mydata\result.txt

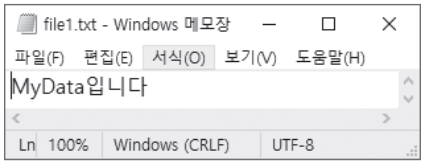
Q3 다음은 영문과 한글이 혼합된 문자열을 MS949와 UTF-8문자셋을 기준으로 byte[]로 변환한 예다. 다음 코드의 실행 결과를 쓰시오(단, 예외 처리는 고려하지 않음).

```
byte[] a = "abc가나다".getBytes("MS949");
byte[] b = "abc가나다".getBytes("UTF-8");

System.out.println(a.length);
System.out.println(b.length);
```

9
12

Q4 다음과 같이 영문과 한글이 혼용된 텍스트를 UTF-8로 인코딩한 파일을 현재 작업 폴더 위치에 넣어 뒀다.



이때 다음과 같이 FileInputStream 객체를 생성하고, read(byte[] b)를 반복적으로 읽어 리턴되는 count 값을 출력했다. 실행 결과를 쓰시오(단, 예외 처리는 고려하지 않음).

```
File file = new File("file1.txt");
InputStream is = new FileInputStream(file);

byte[] byteArray = new byte[6];                                    6
                                                                   6
                                                                   3
int count;
while ((count = is.read(byteArray)) != -1) {
    System.out.println(count);
}
```

Q5 다음은 "안녕하세요" 문자열을 byte[]로 변환한 후 이를 다시 문자열로 변환하는 코드로, 이 과정에서 byte[]의 일부분만을 문자열로 변환했다. 실행 결과가 "하세요"만 나오도록 byte[] → 문자열 변환 코드를 완성하시오(단, 예외 처리는 고려하지 않음).

```
byte[] a = "안녕하세요".getBytes("UTF-8");
String b = new String(a, 6, 9, "UTF-8"); 또는 new String(a, 6, 9); (디폴트가 UTF-8인 경우)

System.out.println(b); // 하세요
```

실행 결과

```
하세요
```

Q6 다음은 기본값 문자셋이 UTF-8로 설정돼 있을 때 OutputStream의 write() 메서드와 flush() 메서드를 이용해 콘솔에 "반가워"가 출력되도록 코드를 완성하시오(단, 예외 처리는 고려하지 않음).

```
OutputStream os = System.out;

byte[] byteArray = "반가워".getBytes("UTF-8");
os.write(byteArray);
os.flush();
```

실행 결과

```
반가워
```

Q7 다음은 작업 폴더에 file2.dat 파일을 작성하는 예제다. 이때 속도 향상을 위해 내부 버퍼를 사용하고, 다양한 타입의 값을 파일에 쓸 수 있는 객체를 생성하시오(이때 참조 변수명은 임의로 지정할 수 있으며, 예외 처리를 고려하지 않음).

```
File file = new File("file2.dat");

FileOutputStream fos = new FileOutputStream(file);
BufferedOutputStream bos = new BufferedOutputStream(fos);
DataOutputStream dos = new DataOutputStream(bos);
```

Q8 다음은 PrintStream 객체를 이용해 콘솔에 출력하는 예다. 실행 결과를 쓰시오.

```
try (OutputStream os = System.out;
    PrintStream ps = new PrintStream(os)) {
    ps.print("안녕");
    ps.print("abc" + "방가" + "\n");
    ps.printf("%s ", "땡큐").printf("%f %d", 3.5, 7);
} catch (IOException e) {}
```

안녕abc방가
땡큐 3.500000 7

Q9 다음은 char 단위의 파일 출력 클래스인 FileWriter를 이용해 각각 영문과 한글을 저장하고 있는 파일을 생성하는 예제다. 기본값 문자셋이 "MS949"일 때 생성되는 2개 파일(file3.txt, file4.txt)의 인코딩 문자셋을 쓰시오(단, 예외 처리는 고려하지 않음).

```
File file3 = new File("file3.txt");
try(Writer writer = new FileWriter(file3)) {
    writer.write("Hi\n".toCharArray());
    writer.flush();
}
File file4 = new File("file4.txt");
try(Writer writer = new FileWriter(file4)) {
    writer.write("반갑습니다\n".toCharArray());
    writer.flush();
}
```

파일명	인코딩 문자셋
file3.txt	**UTF-8**
file4.txt	**MS949(ANSI)**

Q10 다음은 byte 단위로 입력받는 콘솔을 이용해 한글을 입력받아 그대로 출력하는 코드다. 기본값 문자셋이 UTF-8로 설정돼 있을 때 다음과 같이 입력된 한글이 깨짐 없이 출력되도록 코드를 완성하시오(단, 예외 처리는 고려하지 않음).

```
Reader reader = new InputStreamReader(System.in, "UTF-8");
int data;
while ((data = reader.read()) != '\r') {
    System.out.print((char) data);
}
```

실행 결과

```
안녕하세요
안녕하세요
```