2008-02-06

2007年度（平成19年度）計算機言語II定期試験問題

I. プログラミング言語Java

1. Java言語で書かれたプログラムを実行するには javac と java という2つのコマンドを使用しなければならない。これらの役割について説明せよ。（10点）
2. Java言語では、クラス記述の中にフィールド、メソッド、クラスフィールド、クラスメソッドを書くことができる。これらの違いについて説明せよ。（10点）
3. 次の2つの代入文の際に起こることを対比しながら、代入に関する基本型と参照型の違いについて説明せよ。（15点）

```
float f = 1;
Reader r = new FileReader(“sample.txt”);
```

II. Javaプログラミング

次のプログラムをJavaで実装せよ。必要なら、別途配布するAPIマニュアルに記載されているクラスやメソッドを用いて構わない。

1. 自然数nの約数をすべて求めるクラス Divisor。nはコマンドラインの引数で与える。実行例は以下のようになる。（20点）

```
$ java Divisor 200
1
2
4
5
8
10
20
25
40
50
100
200
```

2. 複数のファイルに対し、指定された文字列を含む行とその行番号を出力するクラスGrep。実行例は以下のようになる。この例では、2つのファイル compiler20060726.tex, database-2000726.tex に対し、”latex”という文字列が含まれている行とその行番号を出力している。出力フォーマットは「ファイル名:行番号:行の内容」とする。（30点）

```
$ java Grep latex compiler20060726.tex database-20060727.tex
compiler20060726.tex:1:%#!platex
database-20060727.tex:32:レポートはlatexで書くこと
```

III. XML

以下に示すHTML文書は、XMLとしてみると文法的に正しくない。どこが正しくないか指摘せよ。（15点）

```
<head>
<title>自己紹介</title>
</head>
<body bgcolor=white text=black>
<h1>県大太郎</h1>
<img alt='写真' src='myphoto.jpg'>
生まれも育ちも総社です。今年で20歳になりました。
<p>
今は<a href="http://www.oka-pu.ac.jp/">岡山県立大学</a>で学生をしています。
<p>
毎日のように実験とレポートに追われています。しんどいです。
<ul>
<li>出身高校…岡山県立大学附属高校
<li>大学…岡山県立大学情報工学部情報通信工学科
<li>趣味…パソコン自作
<li>大学の仲間とバンドを組んでいます。学園祭に出てますので、よかったら見に来てください。
</ul>
<hr>
最終更新日：2008年2月6日
</body>
```

Java SE APIマニュアル（一部抜粋、表現の部分的な変更あり）

クラスの説明中、*java.lang.A* → *java.lang.B* というような記法を用いている。これはクラス java.lang.B がクラス java.lang.A を継承していることを表す。

クラスInteger（java.langパッケージ）

java.lang.Object → *java.lang.Number* → *java.lang.Integer*

クラスメソッド

public static int parseInt(String s)
文字列の引数を符号付き 10 進数の整数型として構文解析します。文字列にある文字はすべて、10 進数でなければなりません。ただし、1 番目の文字だけは、負の値を表すためにマイナス記号の ASCII 文字「-」(「\u002d」) であってもかまいません。以上の結果生成された整数値が返されます。
パラメータ:
　s - 構文解析対象の int 表現を含む String
戻り値:
　10 進数の引数で表される整数値

クラスString（java.langパッケージ）

java.lang.Object → *java.lang.String*

コンストラクタ

public String(char[] value)
新しい String を割り当てて、これが文字配列引数に現在含まれている文字シーケンスを表すようにします。文字配列の内容がコピーされます。コピー後に文字配列が変更されても、新しく作成された文字列には影響しません。
パラメータ:
　value - 文字列の初期値

メソッド

public int length()
この文字列の長さを返します。長さは文字列内の 16 ビット Unicode 文字の数に等しくなります。
戻り値:
　このオブジェクトによって表される文字シーケンスの長さ

public char charAt(int index)
指定されたインデックス位置にある char 値を返します。インデックスは、0 から length() - 1 の範囲になります。配列のインデックス付けの場合と同じように、シーケンスの最初の char 値のインデックスは 0、次の文字のインデックスは 1 と続きます。
パラメータ:
　index - char 値のインデックス
戻り値:
　文字列内の指定されたインデックス位置にある char 値。最初の char 値のインデックスが 0 になる

例外:
　java.lang.IndexOutOfBoundsException - index 引数が負の値、または文字列の長さ以上である場合

public boolean equals(Object anObject)
この文字列と指定されたオブジェクトを比較します。引数が null でなく、このオブジェクトと同じ文字シーケンスを表す String オブジェクトである場合にだけ、結果は true になります。
パラメータ:
　anObject - この String と比較されるオブジェクト
戻り値:
　String が等しい場合は true、そうでない場合は false

public int compareTo(String anotherString)
2 つの文字列を辞書的に比較します。比較は文字列内のそれぞれの文字の Unicode 値に基づいて行われます。この String オブジェクトによって表される文字シーケンスが、引数文字列によって表される文字シーケンスと辞書的に比較されます。この String オブジェクトが辞書的に引数文字列より前にある場合は、結果は負の整数になります。この String オブジェクトが辞書的に引数文字列の後ろにある場合、結果は正の整数になります。文字列が等しい場合、結果は 0 になります。equals(Object) メソッドが true を返すとき、compareTo は 0 を返します。
パラメータ:
　anotherString - 比較対象の String
戻り値:
　引数文字列がこの文字列に等しい場合は、値 0。この文字列が文字列引数より辞書的に小さい場合は、0 より小さい値。この文字列が文字列引数より辞書的に大きい場合は、0 より大きい値

public int indexOf(int ch)
この文字列内で、指定された文字が最初に出現する位置のインデックスを返します。値 ch を持つ文字がこの String オブジェクトによって表される文字シーケンス内にある場合、最初に出現する位置のインデックス (Unicode コード単位) が返されます。
パラメータ:
　ch - 文字 (Unicode コードポイント)
戻り値:
　このオブジェクトによって表される文字シーケンス内で、指定された文字が最初に出現する位置のインデックス。文字がない場合は -1

public int indexOf(String str)
この文字列内で、指定された部分文字列が最初に出現する位置のインデックスを返します。返される整数は、this.startsWith(str, k) が true となるような最小値 k です。
パラメータ:
　str - 任意の文字列
戻り値:
　文字列引数がこのオブジェクト内の部分文字列である場合は、該当する最初の部分文字列の最初の文字のインデックス。部分文字列がない場合は -1

public int lastIndexOf(int ch)
この文字列内で、指定された文字が最後に出現する位置のインデックスを返します。
パラメータ:
　ch - 文字 (Unicode コードポイント)
戻り値:

このオブジェクトによって表される文字シーケンス内で、指定された文字が最後に出現する位置のインデックス。文字がない場合は -1

public int lastIndexOf(String str)
この文字列内で、指定された部分文字列が一番右に出現する位置のインデックスを返します。返されるインデックスは、this.startsWith(str, k) 式が true となるような最大値 k です。
パラメータ:
str - 検索対象の部分文字列
戻り値:
文字列引数がこのオブジェクト内の部分文字列として 1 回以上出現する場合は、該当する最後の部分文字列の最初の文字のインデックス。部分文字列として出現しない場合は、-1

public String substring(int beginIndex)
この文字列の部分文字列である新しい文字列を返します。部分文字列は指定されたインデックスで始まり、この文字列の最後までになります。
パラメータ:
beginIndex - 開始インデックス (この値を含む)
戻り値:
指定された部分文字列
例外:
java.lang.IndexOutOfBoundsException - beginIndex が負の値の場合、あるいはこの String オブジェクトの長さより大きい場合

public String substring(int beginIndex, int endIndex)
この文字列の部分文字列である新しい文字列を返します。部分文字列は、指定された beginIndex から始まり、インデックス endIndex - 1 にある文字までです。したがって、部分文字列の長さは endIndex-beginIndex になります。
パラメータ:
beginIndex - 開始インデックス (この値を含む)
endIndex - 終了インデックス (この値を含まない)
戻り値:
指定された部分文字列
例外:
IndexOutOfBoundsException - beginIndex が負の値である場合、endIndex がこの String オブジェクトの長さより大きい場合、あるいは beginIndex が endIndex より大きい場合

public String replace(char oldChar, char newChar)
この文字列内にあるすべての oldChar を newChar に置換した結果生成される、新しい文字列を返します。
文字 oldChar がこの String オブジェクトによって表される文字列内にない場合は、この String オブジェクトへの参照が返されます。そうでない場合は、この String オブジェクトによって表される文字列と同じ文字列を表す、新しい String オブジェクトが生成されます。ただし、文字列内の oldChar はすべて newChar に置換されます。
パラメータ:
oldChar - 以前の文字
newChar - 新しい文字
戻り値:
この文字列内のすべての oldChar を newChar に置換することによって生成された文字列

public String toLowerCase()
この String 内のすべての文字を小文字に変換します。
戻り値:
　小文字に変換される String

public String toUpperCase()
この String 内のすべての文字を大文字に変換します。
戻り値:
　大文字に変換された String

public String trim()
文字列のコピーを返します。先頭と最後の空白は省略されます。
このメソッドは文字列の先頭と最後から (上記で定義された) 空白を切り取るために使用できます。
戻り値:
　この文字列の先頭と最後の空白を削除したコピー、またはこの文字列 (先頭と最後に空白が存在しない場合)

クラスFileReader（java.ioパッケージ）

java.lang.Object → java.io.Reader → java.io.InputStreamReader → java.io.FileReader

コンストラクタ

FileReader(File file)
読み込み元の File を指定して、新規 FileReader を作成します。
パラメータ:
　file - 読み込み元の File
例外:
　java.io.FileNotFoundException - ファイルが存在しないか、普通のファイルではなくディレクトリであるか、または何らかの理由で開くことができない場合

FileReader(String fileName)
読み込み元のファイルの名前を指定して、新規 FileReader を作成します。
パラメータ:
　fileName - 読み込み元のファイルの名前
例外:
　java.io.FileNotFoundException - ファイルが存在しないか、普通のファイルではなくディレクトリであるか、または何らかの理由で開くことができない場合

メソッド

public void close()
ファイルリーダを閉じます。
例外:
　java.io.IOException - 入出力エラーが発生した場合

public int read()
単一の文字を読み込みます。

戻り値:
　読み込まれた文字。ストリームの終わりに達した場合は -1
例外:
　java.io.IOException - 入出力エラーが発生した場合

クラスBufferedReader（java.ioパッケージ）

java.lang.Object → java.io.Reader → java.io.BufferedReader

コンストラクタ

public BufferedReader(Reader in)
デフォルトサイズのバッファでバッファリングされた、文字型入力ストリームを作成します。

メソッド

public int read()
単一の文字を読み込みます。
戻り値:
　0 ～ 65535 (0x00-0xffff) の範囲の整数としての、読み込まれた文字。ストリームの終わりに達した場合は -1
例外:
　java.io.IOException - 入出力エラーが発生した場合

public String readLine()
1 行のテキストを読み込みます。1 行の終端は、改行 (「\n」) か、復帰 (「\r」)、または復行とそれに続く改行のどれかで認識されます。
戻り値:
　行の内容を含む文字列、ただし行の終端文字は含めない。ストリームの終わりに達している場合は null
例外:
　java.io.IOException - 入出力エラーが発生した場合

public void close()
ストリームを閉じます。
例外:
　java.io.IOException - 入出力エラーが発生した場合

クラスFile（java.ioパッケージ）

(注) 以下の説明にある「抽象パス名」とは、Fileオブジェクトと解釈して構わない。
java.lang.Object → java.io.File

コンストラクタ

public File(String pathname)
指定されたパス名文字列から新しい File のインスタンスを生成します。
パラメータ:
　pathname - パス名文字列
例外:

java.lang.NullPointerException - pathname 引数が null の場合

メソッド

public String getName()
この抽象パス名が示すファイルまたはディレクトリの名前を返します。これは、パス名の名前シーケンスの最後の名前です。パス名の名前シーケンスが空の場合、空の文字列が返されます。
戻り値:
この抽象パス名が示すファイルまたはディレクトリの名前。このパス名の名前シーケンスが空の場合は空の文字列

public String getPath()
この抽象パス名をパス名文字列に変換します。結果の文字列は、システムに依存するデフォルトの名前区切り文字（注：Unixだと '/'）を使用して名前シーケンスの名前を区切ります。
戻り値:
この抽象パス名の文字列形式

public String getAbsolutePath()
この抽象パス名の絶対パス名文字列を返します。
この抽象パス名がすでに絶対である場合、パス名文字列は getPath() メソッドのように簡単に返されます。この抽象パス名が空の抽象パス名の場合、現在のユーザディレクトリのパス名文字列が返されます。そうでない場合、このパス名はシステムに依存する方法で解決されます。UNIX システムの場合、相対パス名は現在のユーザディレクトリを基準に解決することで絶対になります。
戻り値:
この抽象パス名と同じファイルまたはディレクトリを示す絶対パス名文字列
例外:
java.lang.SecurityException - 必須のシステムプロパティの値にアクセスできない場合

public boolean exists()
この抽象パス名が示すファイルまたはディレクトリが存在するかどうかを判定します。
戻り値:
この抽象パス名が示すファイルまたはディレクトリが存在する場合は true、そうでない場合は false
例外:
java.lang.SecurityException - ファイルまたはディレクトリへの読み込みアクセスを許可しない場合

public boolean createNewFile()
この抽象パス名が示す空の新しいファイルを生成します (そのファイルがまだ存在しない場合だけ)。
戻り値:
指定されたファイルが存在せず、ファイルの生成に成功した場合は true、示されたファイルがすでに存在する場合は false
例外:
java.io.IOException - 入出力エラーが発生した場合
java.lang.SecurityException - ファイルへの書き込みアクセスを許可しない場合

public boolean delete()
この抽象パス名が示すファイルまたはディレクトリを削除します。このパス名がディレクトリを示す場合、そのディレクトリが削除されるためには空である必要があります。
戻り値:

ファイルまたはディレクトリが正常に削除された場合は true、そうでない場合は false
例外:
java.lang.SecurityException - ファイルへの削除アクセスを許可しない場合

public String[] list()
この抽象パス名が示すディレクトリにあるファイルおよびディレクトリを示す文字列の配列を返します。
この抽象パス名がディレクトリを示さない場合、このメソッドは null を返します。ディレクトリを示す場合は、文字列の配列が返されます。文字列は、ディレクトリ内の各ファイルまたはディレクトリごとに 1 つです。そのディレクトリ自体およびその親のディレクトリを示す名前は結果に含まれません。各文字列は、絶対パスではなくファイル名です。
結果の配列の名前文字列は特定の順序にはなりません。アルファベット順になるわけではありません。
戻り値:
この抽象パス名が示すディレクトリにあるファイルおよびディレクトリを示す文字列の配列。ディレクトリが空の場合、配列は空になる。この抽象パス名がディレクトリを示さない場合、または入出力エラーが発生した場合は null
例外:
java.lang.SecurityException - ディレクトリへの読み込みアクセスを許可しない場合

public boolean mkdir()
この抽象パス名が示すディレクトリを生成します。
戻り値:
ディレクトリが生成された場合は true、そうでない場合は false
例外:
java.lang.SecurityException - 指定されたディレクトリの生成を許可しない場合

public boolean mkdirs()
この抽象パス名が示すディレクトリを生成します。存在していないが必要な親ディレクトリも一緒に作成されます。このオペレーションが失敗した場合でも、いくつかの必要な親ディレクトリの生成には成功している場合があります。
戻り値:
必要なすべての親ディレクトリを含めてディレクトリが生成された場合は true、そうでない場合は false
例外:
java.lang.SecurityException - 指定されたディレクトリと、必要なすべての親ディレクトリの生成を許可しない場合

public boolean renameTo(File dest)
この抽象パス名が示すファイルの名前を変更します。
パラメータ:
dest - 指定されたファイルの新しい抽象パス名
戻り値:
名前の変更が成功した場合は true、そうでない場合は false
例外:
java.lang.SecurityException - 古いパス名と新しいパス名のどちらかへの書き込みアクセスを許可しない場合
java.lang.NullPointerException - パラメータ dest が null の場合