整数型

public class data1java {

public static void main(String[] args) {

System.**out**.println(100);

System.**out**.println(0);

System.**out**.println(-123);

Javaの整数型は４種類ある

byte 8bit

short 16bit

int 32bit

long 64bit

bit=桁数　なので8bitは8桁という認識

値の最大値を越える場合はエラーとなる

System.out.println(2147483648);

例えばこの場合は値の最大値を越えるためエラーとなる

こういった場合は末にＬを付けるようにする

浮動小数点

public class data2java {

public static void main(String[] args) {

System.**out**.println(0.1);

System.**out**.println(1.618);

System.**out**.println(-9876.0123);

}

}

Javaの浮動小数点型は二種類ある

float 32bit

double 64bit

とりあえずdoubleはデカいという認識でおｋ

浮動小数点のリテラルはdouble扱いになる

デカいので困ることはまずない

論理型 boolean

public static void main(String[] args) {

System.**out**.println(true);

System.**out**.println(false);

}

}

Javaの論理型はboolean型の事、boolean型のみで他に種類はない

１バイトだが環境によって違うらしい

文字・文字列型

public class data4 {

public static void main(String[] args) {

System.**out**.println("こんにちは");

// 一文字だけの文字列

System.**out**.println('s');

}

Javaの文字列型はString型の事を指す、他のデータ型と毛色が異なる

String型は後ほど開発

なお１文字だけの文字は「’’」のようにシングルクォーテーションで囲む

１文字はあくまで「文字」で「文字列」ではない

言語によっては文字列もシングルクォーテーションで囲めるが

Javaではエラーになる

Javaで１文字だけを扱うときはchar型を使う、charの値は文字コード（文字の識別番号）

たとえば「12354」という値の文字コードが「あ」になる

算術演算子、整数のみでの計算

public class data5 {

public static void main(String [] args) {

// 簡単な計算をする

System.**out**.println(1+1);

}

}

Javaの代表的な算術演算子

+　加算

-　減産

\*　乗算

/　除算

%　余剰演算

算術演算子を使って色々計算してみる

public class data6 {

public static void main(String[] args) {

// 加算

System.**out**.println(45 + 18);

// 減算

System.**out**.println(30 - 12);

// 乗算

System.**out**.println(15 \* 6);

// 除算

System.**out**.println(30 / 3);

// 余剰演算

System.**out**.println(45 % 8);

}

}

整数・小数が混在する計算

public class data7 {

public static void main(String[] args) {

System.**out**.println(45 + 1.8);

System.**out**.println(3 - 1.2);

System.**out**.println(15 \* 0.6);

System.**out**.println(3 / 0.3);

System.**out**.println(45 % 0.8);

}

}

実行結果からして、通常は45を0.8で割った余りは0.2だけど

結果に若干のズレがある

これは浮動小数点型は誤差が生じる場合があって

そもそも0.8はプログラム上は「0.8のおおよその数値」として扱われる

厳密には0.8ではないため「45 % 0.8」の結果も少しズレてしまう。

例えば99.9という少数はコンピューター内部的には99.9000000000000　のような

おおよそ数値となっている

要はコンピューターは少数の扱いが苦手ということ

消費税の計算など正確な結果が求められる場面では、浮動小数点型をそのまま使う事はできない

誤差を切り捨てるなどといった専用の処理が必要になってくる

多少の誤差が書字ても問題ない場面（大まかな平均値が必要な場合など）では、浮動小数点は便利

浮動小数点型は誤差が生じる場合があるという認識でおｋ

文字列の連結

public class data8 {

public static void main(String[] args) {

// 文字列の連結

System.**out**.println("テスト" + "テスト");

// 整数と文字列の連結

System.**out**.println("テスト" + 123);

}

}

文字列のデータ同士は+を使うことで連結できる

Javaでは片方のデータが数値（整数や少数）でも連結可能

数値を文字列型に変換したうえで連結する

複数の計算や連結を一度に行う場合、左から順に処理されるのが基本

この性質を理解しておく事