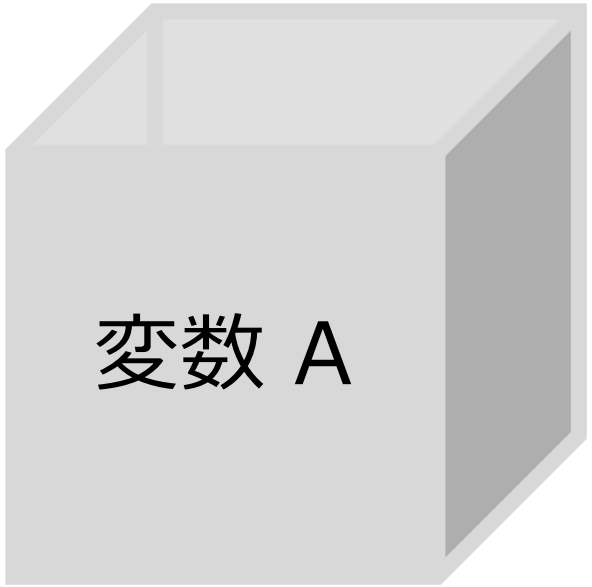
Java

変数と代入

4時間目

変数（variable）とは？

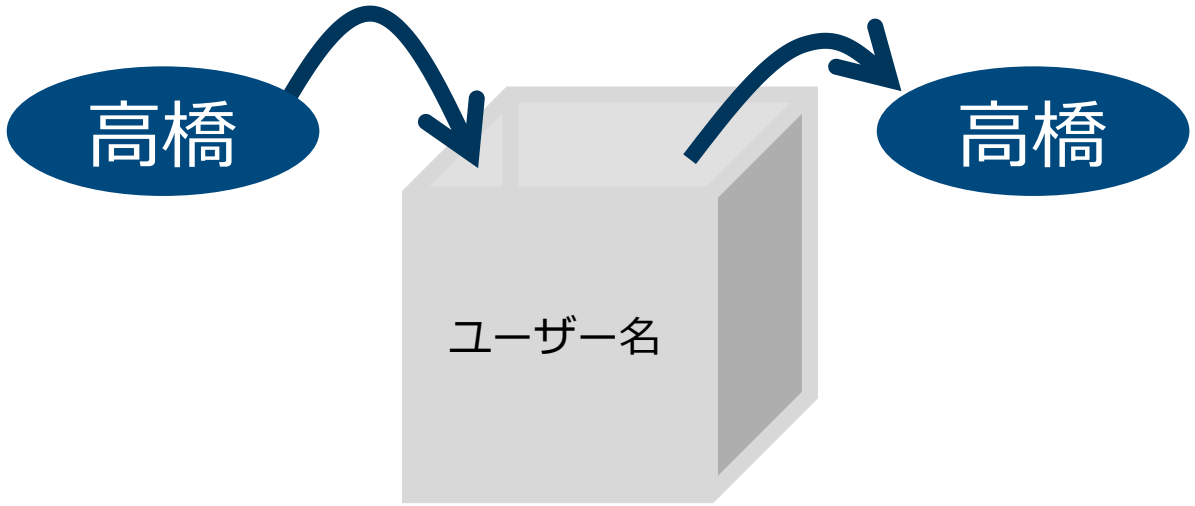
あとで、何かしらの値が入る領域のこと



何に使うの？

値を入れたり、出して使ったり。

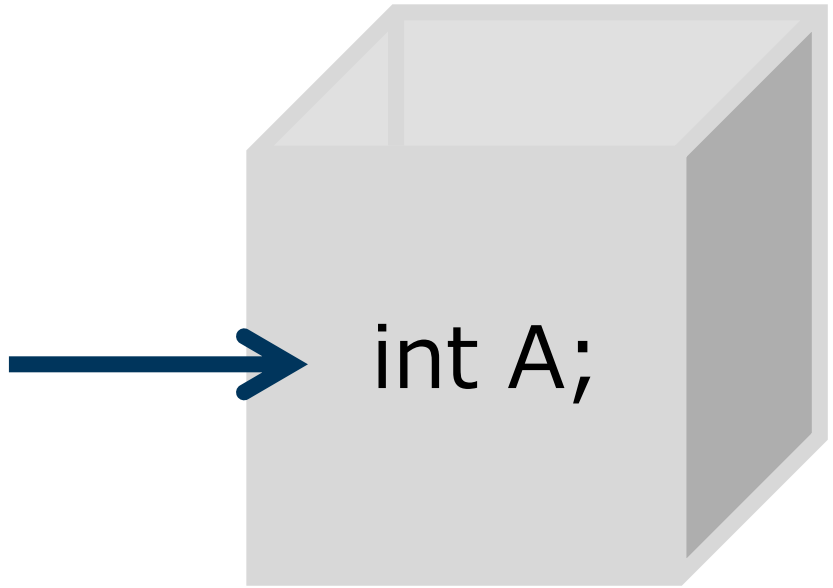
プログラムに広がりが出る。



補足　具体的な使い方は後日たくさん出てきます。まずは、取り扱いの基本を覚えましょう。

変数の定義

変数を使う前に、型と名前の定義が必要



プリミティブ型（基本的な型）の種類

型の種類	型の名前	解説
論理値型	boolean	真偽値（true または false）を格納できる
文字型	char	16bitのUnicode文字（'¥u0000'～'¥uffff'）を格納できる
整数型	byte	8bitの符号付整数（-128～127）を格納できる
	short	16bitの符号付整数（-32768～32767）を格納できる
	int	32bitの符号付整数（-2147483648～2147483647）を格納できる
	long	64bitの符号付整数（-9223372036854775808 ～9223372036854775807）を格納できる
浮動小数点型	float	32bitの単精度浮動小数点数を格納できる
	double	64bitの倍精度浮動小数点数を格納できる

文字列を格納できる型は？

文字列を格納できるプリミティブ型はない

文字とは、「あ」「い」「う」など1つひとつを意味し、
文字列とは、「単語」「文章」などの文字の集まりを意味します。

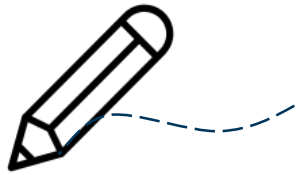
Stringクラスを参照して、型を定義する

補足

Stringクラスは、“java.lang”というパッケージの中にあるAPIで、Javaにおける APIとは、クラスライブラリ（共通で使用できる部品）のことです。ここでは、「API」「クラスライブラリ」などの意味はわからなくて大丈夫です。

クイズ

変数を探してください



```
int suugaku = 45;
int eigo = 82;
if (suugaku > 50 || eigo > 50){
        System.out.println( "この人は合格です" );
}
```

補足

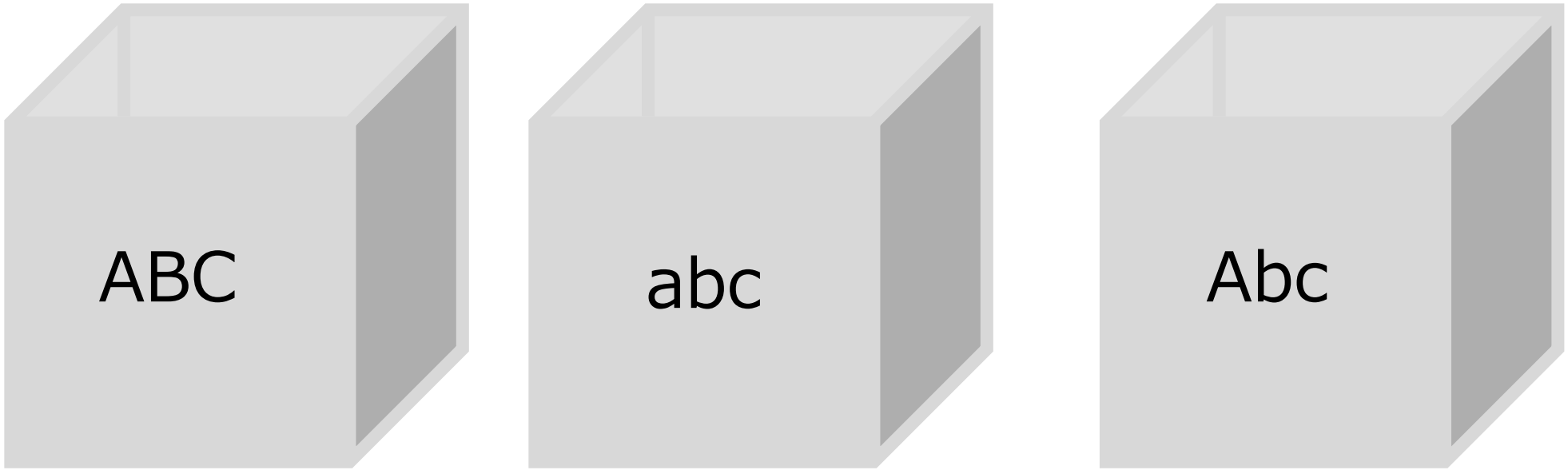
このプログラムの意味はわからなくて大丈夫です。

答えあわせ

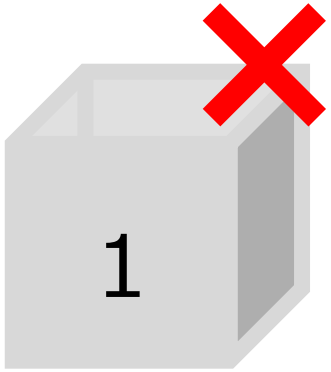
```
int suugaku = 45;
int eigo = 82;
 if (suugaku > 50 || eigo > 50){
        System.out.println( "この人は合格です" );
}
```

大文字と小文字は区別される

すべて別の変数

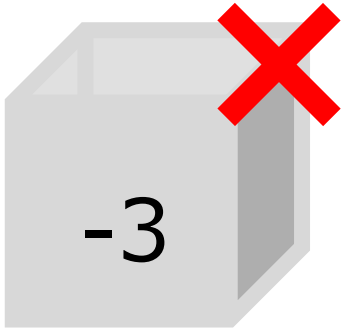


変数名に使えないもの



数字

ただし、
アルファベットのうしろにつく場合は良い
（ a2、_d555 など）

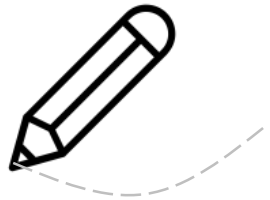


「 _ 」以外の記号

アンダーバー

クイズ

NGな変数名を「５個」探してください



8	abc	/dk
8924	Xyz	abc-xyz
-3	a2	_d555

クイズ

答えあわせ

✕ 8

数字は **NG**

abc

✕ /dk

「 _ 」以外は **NG**

✕ 8924

数字は **NG**

Xyz

✕ abc-xyz

「 _ 」以外は **NG**

✕ -3

「 _ 」以外は **NG**

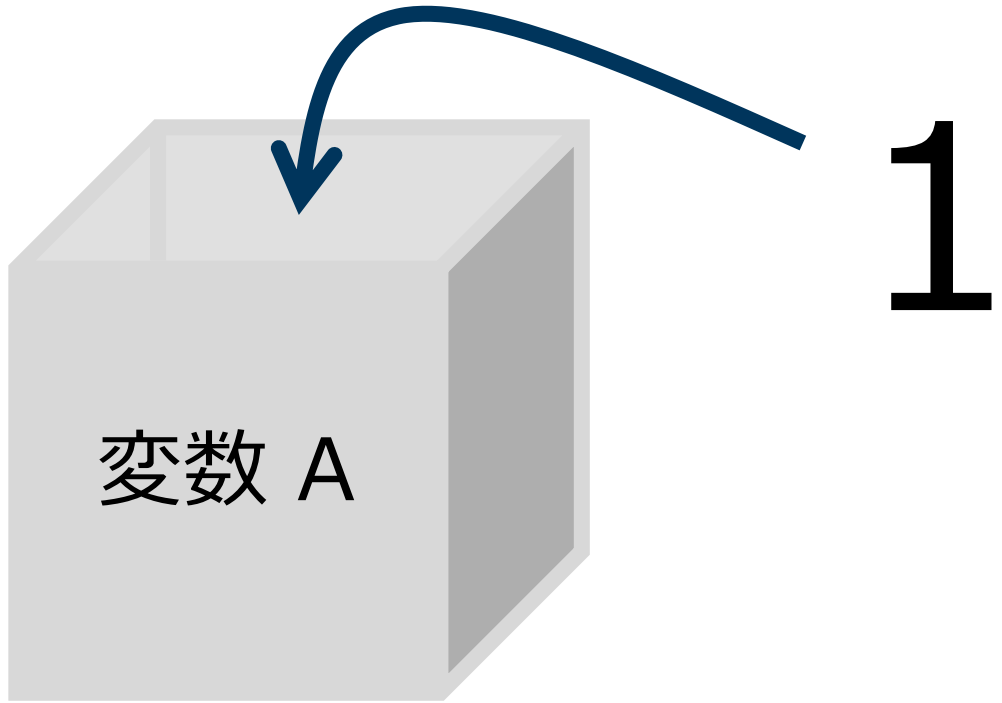
a2

_d555

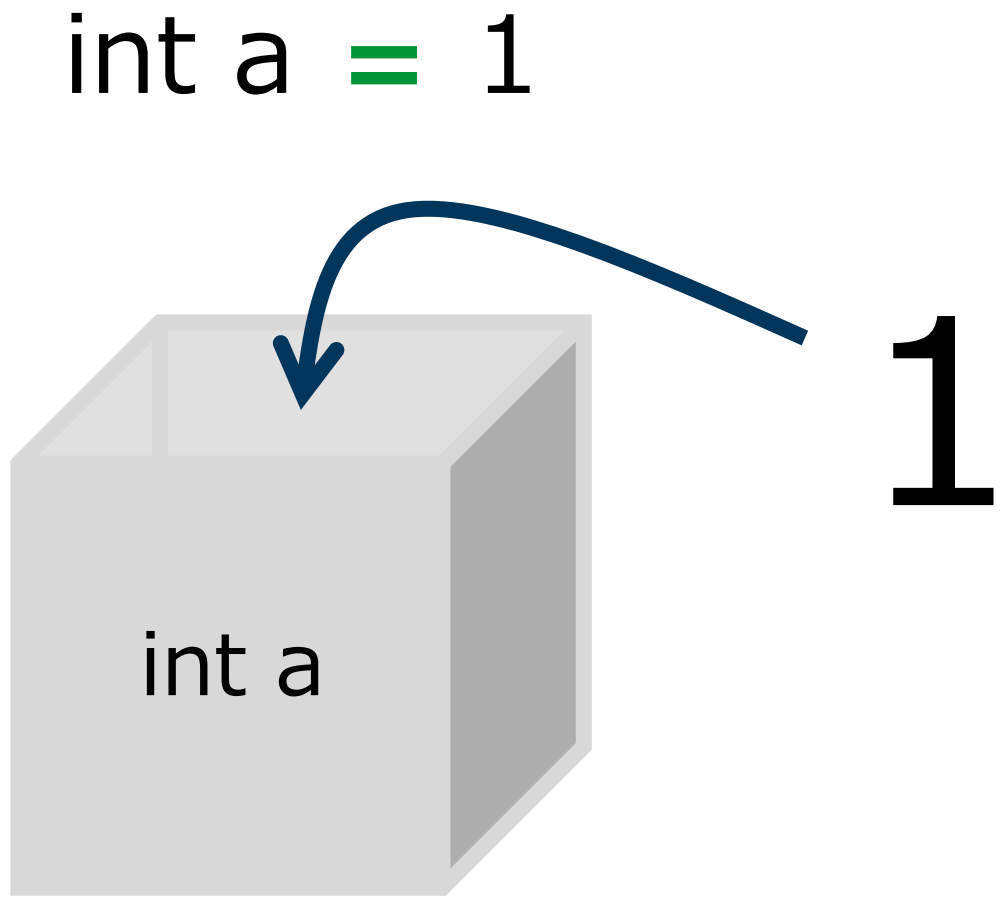
「 _ 」はOK

変数には値を入れることができる

「代入」と言います



代入は「＝」で書く



利用例

解説

例では、変数 name に「高橋」を代入し、name を System.out.println で出力しています。
すると実行結果には、「高橋」と表示されます。

Javaファイル

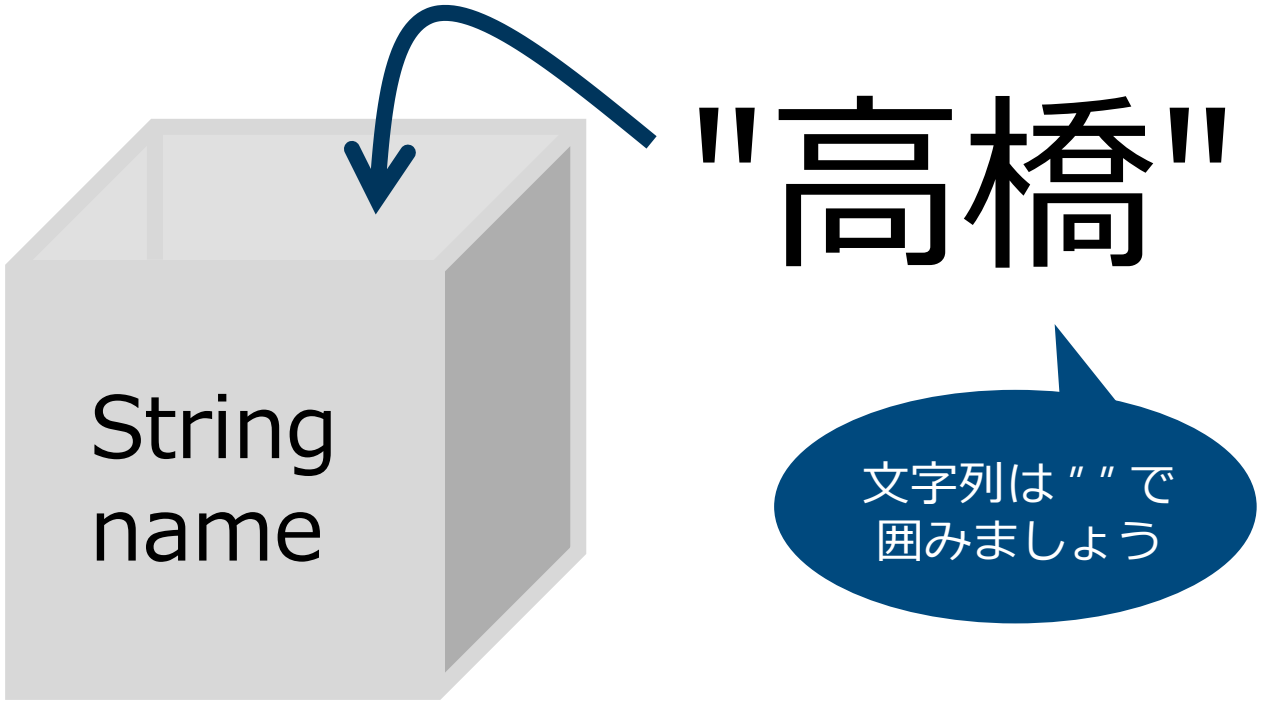
```
String name = "高橋";
System.out.println(name);
```

代入して

出力

実行結果

```
高橋
```



参考　代入のイメージ

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

String str1 = "鈴木";
System.out.println(str1);

        }
}
```

鈴木

参考　代入のイメージ

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

String str2 = “Tanaka";
System.out.println(str2);

        }
}
```

アルファベットも文字列なので、
前のページと同様に処理

Tanaka

参考　代入のイメージ

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

String tel = “090-1234-5678";
System.out.println(tel);

        }
}
```

数字とハイフンがあるので
文字列として処理する

```
090-1234-5678
```

参考　代入のイメージ

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

char char1 = ‘A’;
System.out.println(char1);

        }
}
```

```
A
```

文字1つは、「文字列」でなく「文字」なので、
Stringではなく、charで処理する。
また、charの場合は、代入する値を、必ず
シングルクオテーションで囲まなければいけません。

参考

代入のイメージ

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

int num1 = 12345;
System.out.println(num1);

        }
}
```

数字は、intで処理をする

```
12345
```

参考 代入のイメージ

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

boolean boo= true;
System.out.println(boo);

        }
}
```

真偽値は、booleanで処理をする

```
true
```

囲み文字が不要な場合

解説

計算式や数字を代入するときは、囲み文字は不要です。
論理値型でも同様に、囲み文字なしで true または false を代入します。

Javaファイル

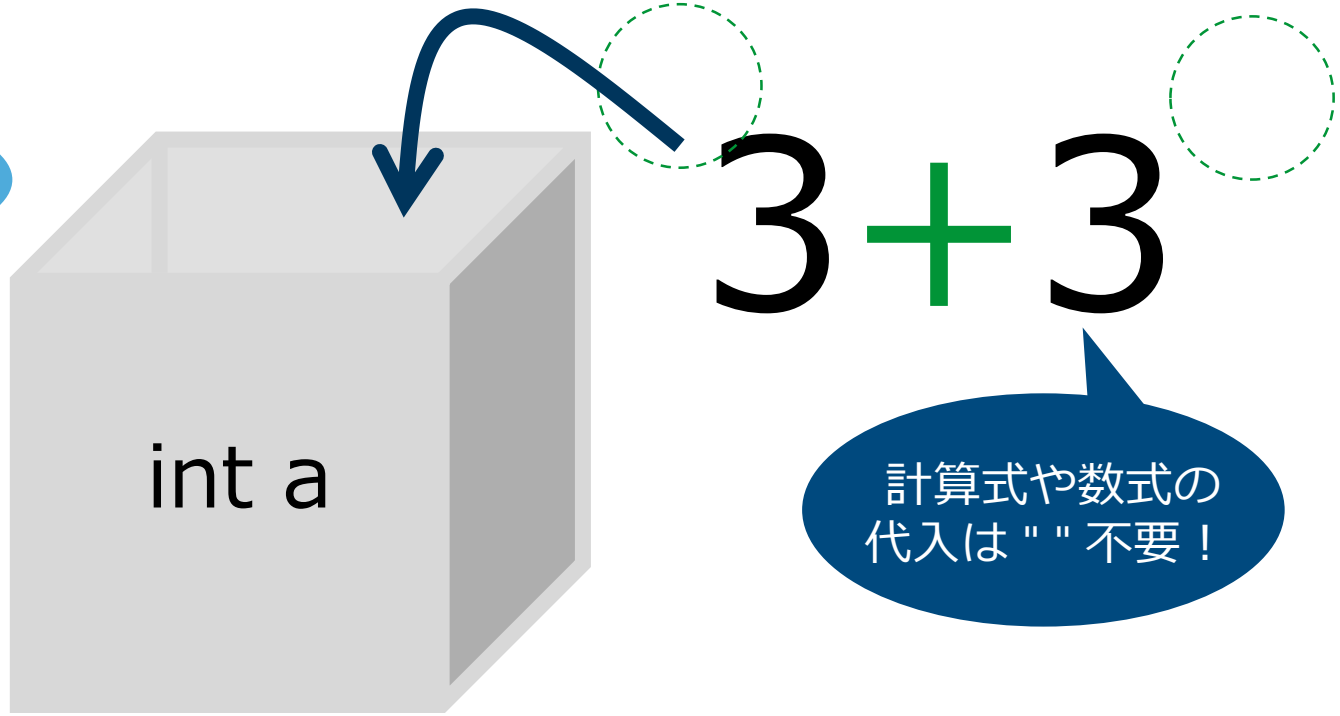
```
int a = 3+3;
System.out.println(a);
```

計算式として代入

実行結果

```
6
```

計算結果が表示される



囲み文字に " " を付ける場合

解説

文字列型（String）で定義した場合、値は " "（ダブルクォーテーション）で囲みます。
囲まない場合、エラーになります。

Javaファイル

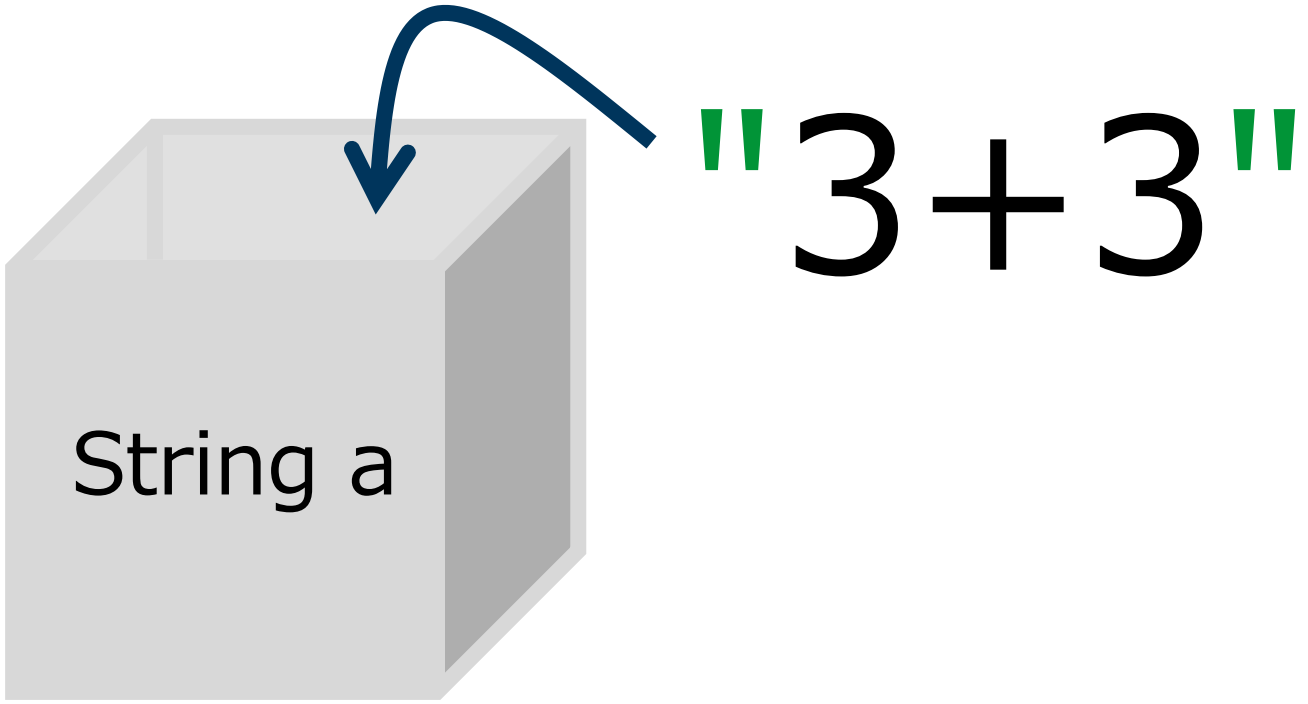
```
String a = "3+3";
System.out.println(a);
```

文字列を代入

実行結果

```
3+3
```

そのまま表示



囲み文字に ' ' を付ける場合

解説

文字型（char）で定義した場合、値は‘ ’（シングルクオテーション）で囲みます。
囲まない場合、エラーになります。

Javaファイル

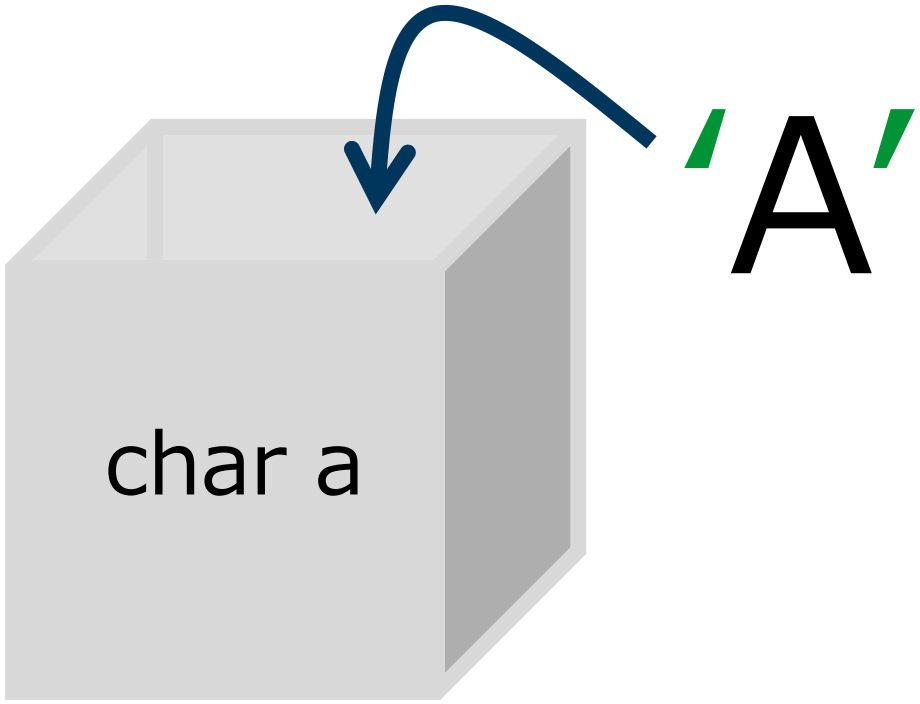
```
char a = ‘A’;
System.out.println(a);
```

1 文字を代入

実行結果

```
A
```

そのまま表示



参考　さまざまな実行結果

Javaファイル

```
int a = 2*5;
System.out.println(a);
```

実行結果

```
10
```

Javaファイル

```
String a = "2*5";
System.out.println(a);
```

実行結果

```
2*5
```

```
char a = 'a';
System.out.println(a);
```

```
a
```

```
char a = a;
System.out.println(a);
```

```
エラー
```

```
String a = a;
System.out.println(a);
```

```
エラー
```

```
boolean  a = true;
System.out.println(a);
```

```
true
```

```
String a = "true";
System.out.println(a);
```

```
true
```

```
String a = true;
System.out.println(a);
```

```
エラー
```

参考　文字の連結

解説　変数に代入した値を出力する際に、　+ を使えば、下記のように文字列の連結が可能です。

Javaファイル

```
public class クラス名 {
        public static void main(String[] args) {

String abc = “山田”;
System.out.println(abc + “さん”);

        }
}
```

abcに代入した「山田」と
「さん」を連結できる。

```
山田さん
```