01　初めてのプログラミング

目標

□　標準入出力を使えるようになる.

□　int 型を使えるようになる.

□　四則演算を使えるようになる.

例題

【合計時間】

出典：JOI 2010/2011 予選 A

問題

太郎君は 3 カ所の店を訪ねることを日課にしている．家を出発し，決まった順番で 3 カ所の店を回った後，家に帰る．ときどき，ストップウォッチを使って，各区間を移動するのに何秒かかったかを計り，その秒数を記録する.

ある日の計測結果が与えられたとき，この日の移動時間の合計が何分何秒かを求めるプログラムを作成せよ.

入力

入力は 4 行からなり，1 行に 1 つずつ正の整数が書かれている.

1 行目の整数は家から 1 つ目の店までの移動時間を表す秒数である.

2 行目の整数は 1 つ目の店から 2 つ目の店までの移動時間を表す秒数である.

3 行目の整数は 2 つ目の店から 3 つ目の店までの移動時間を表す秒数である.

4 行目の整数は 3 つ目の店から家までの移動時間を表す秒数である．ただし，与えられる入力データにおいては合計移動時間は 1 分 0 秒以上で 59 分 59 秒以下であることが保証されている.

出力

出力は 2 行からなる．x 分 y 秒（$1 \leqq x \leqq 59$，$0 \leqq y \leqq 59$）のとき，1 行目に x を，2 行目に y を出力せよ.

入力例 1

```
31
34
7
151
```

出力例 1

```
3
43
```

入出力例 1 では，合計時間が $31+34+7+151=223$ 秒つまり，3 分 43 秒である．

入力例 2

```
316
430
643
1253
```

出力例 2

```
44
2
```

入出力例 2 では，合計時間が 2642 秒つまり，44 分 2 秒である．

入力例 3

```
5
10
15
30
```

出力例 3

```
1
0
```

入出力例 3 では，合計時間が 60 秒つまり，1 分 0 秒である．

考察

- 与えられる整数を順に A,B,C,D とおくと，答えは $A+B+C+D$ 秒である．
- 1 分は 60 秒だから，秒数 $A+B+C+D$ を 60 で割って切り捨てた商が答え x である．
- 同様に，秒数 $A+B+C+D$ を 60 で割った余りが答え y である．

解答例

```
#include <bits/stdc++.h>
using namespace std;
int main(){
    int A, B, C, D, sum, x, y;
    cin >> A >> B >> C >> D;
    sum = A + B + C + D;
    x = sum / 60;
    y = sum % 60;
    cout << x << endl << y << endl;
}
```

解説

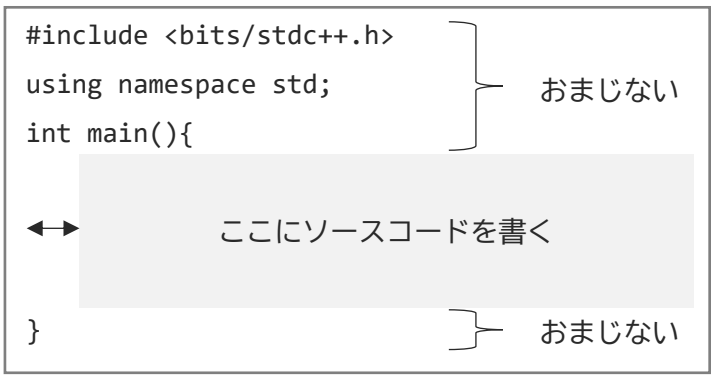
① おまじない

ソースコードの最初と最後の部分

```
#include <bits/stdc++.h>
using namespace std;
int main(){

}
```

はプログラムが正しく動くための「おまじない」である．この部分の意味は難しいので今は気にしなくてよい．解答は `int main(){` と `}` の間に **字下げ**（インデント）して記述する．



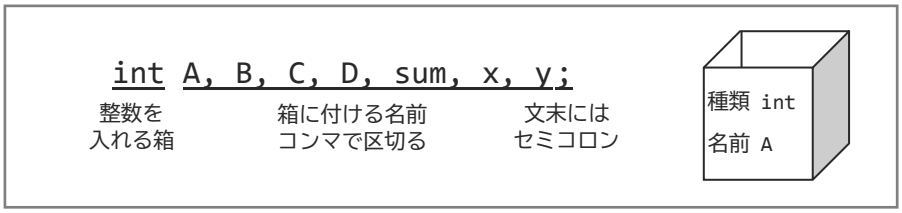
② 変数の定義

プログラムは値を計算するのに **変数** と呼ばれる「箱」を用いる．ソースコードを書く時はまず，変数の定義をすることが多い．変数の定義は次のように行う．

```
int A, B, C, D, sum, x, y;
```

int（イント）は integer（英: 整数）の略であり，その変数が整数を入れるためのものであることを意味する．後に続く `A, B, C, D, sum, x, y` は変数に付ける名前で，コンマ `,` で区切る．変数にはアルファベットと数字からなる名前を自由に付けることができる．行末の `;` は **セミコロン** と呼ばれる記号で，日本語の句点や英語のピリオドと同様に文の終わりを意味する．文の終わりにはセミコロンを必ずつけるようにしよう．

つまりこの文は「整数を入れるための箱を 7 個用意し，それぞれに A，B，C，D，sum，x，y と名前を付けよ」という指示を意味する．

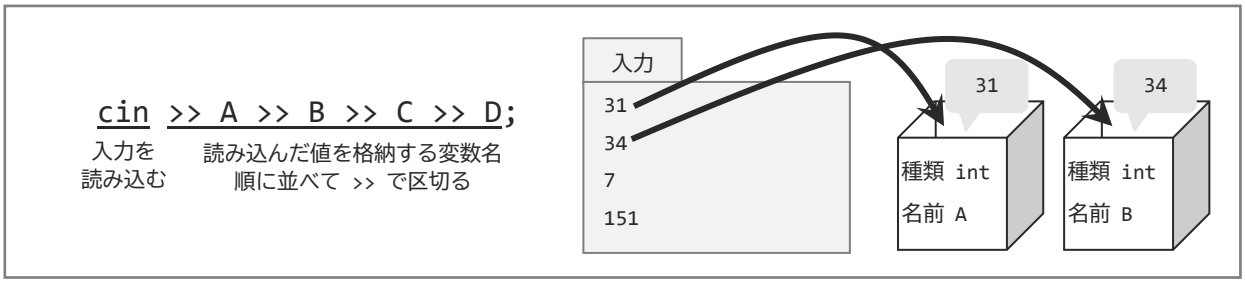


③ 標準入力

キーボードによって入力された数値などを受け取るには **cin**（シーイン）を用いる．

```
cin >> A >> B >> C >> D;
```

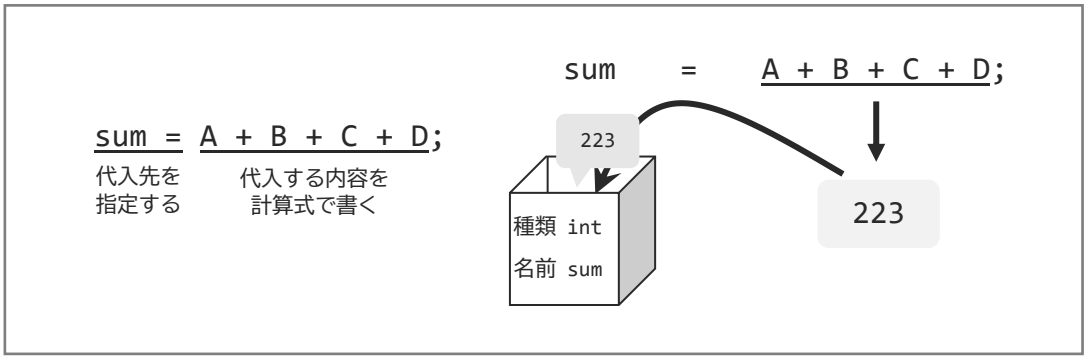
cin >> (変数); の形で記述することで，入力された値を変数に格納できる．今回の解答例のように複数の変数を >> によって繋げて記述することもでき，左に書かれた変数から順に入力が読み込まれる．



④ 四則演算と代入

```
sum = A + B + C + D;
```

(変数) = (計算式); の形で記述することで，右辺の計算結果を左辺の変数に **代入** できる．左辺の結果を右辺に代入することはできないので注意する．



右辺に記述する計算式で使える記号は下表の通りである．

	日本で使われている記号	C++ における記号
加法	+	+
減法	−	-
乗法	×	*（アスタリスク）
除法	÷	/（スラッシュ）
剰余 割り算の余り	mod	%

計算順序は乗法，除法，剰余が優先されるが，かっこをつけて制御できる．

⑤ 不思議な式

なお，例えば次のような表現も可能である．

```
A = A + 1;
```

このとき，A に入った値は 1 大きくなる．= はあくまでも右辺の計算結果を左辺に代入するという意味で，「左辺と右辺が等しい」という等号本来の意味は無い．

⑥ 割り算の結果は切り捨て

記号 / による割り算の結果は切り捨てである．

```
x = sum / 60;
```

この場合，変数 sum を 60 で割った商の整数部分が変数 x に代入される．

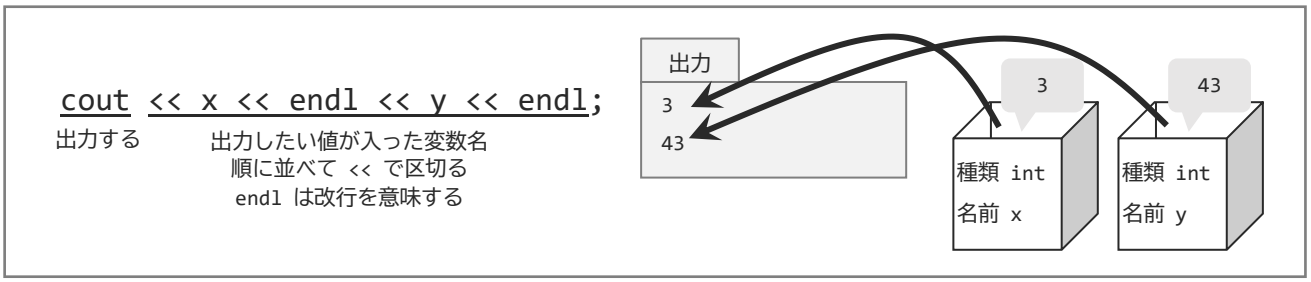
⑦ 標準出力

画面に数値や文字列を表示するには **cout**（シーアウト）を使う．

```
cout << x << endl << y << endl;
```

cout << (式や変数); の形で記述することで，変数の中身を画面に表示させることができる．今回の解答のように複数の変数を << によって繋げて記述することもでき，左のものから順に出力される．cin は >>，cout は << で向きが違うことに注意する．

改行を出力するには endl（end line の略）を使う．これは改行が最初から代入されている変数だと思って今は差し支えない．**問題で特に指示が無くても，最後には必ず改行を出力する．**



なお，半角スペースを出力したいときは次のようにする．この意味は後の章で扱う．

```
cout << " ";
```

表現

（1）標準入力から 2 つの整数を受け取って，その和を出力する

```
int num1, num2;
cin >> num1 >> num2;
cout << num1 + num2 << endl;
```

（2）標準入力から 1 つの自然数を受け取って，それが偶数なら 0，奇数なら 1 を出力する

```
int A;
cin >> A;
cout << A % 2 << endl;
```

（3）例題の異なる解答例

```
int A, B, C, D;
cin >> A >> B >> C >> D;
cout << (A + B + C + D) / 60 << endl;
cout << (A + B + C + D) % 60 << endl;
```

類題

- ☐ box（AtCoder Beginner Contest 180 A）
- ☐ Discount Fare（AtCoder Beginner Contest 113 A）
- ☐ Calc（AtCoder Beginner Contest 172 A）
- ☐ Garden（AtCoder Beginner Contest 106 A）
- ☐ Payment（AtCoder Beginner Contest 173 A）