

# プログラミング基礎 #05

## 反復

担当：向井 智彦

# 前回のおさらい

- bool型
  - true と false のいずれか 真偽値
- 比較演算
  - 数値の等しさ・大小関係をbool型で評価
- 論理演算
  - bool型変数に対する論理和・論理積
- 条件分岐
  - bool値に応じて異なる処理を行う仕組み

# 本日の内容

- 反復 while, for
- 変数のスコープ
- インクリメント++とデクリメント--

@ wandbox.org

```
#include <iostream>
int main()
{
    for (int i = 0; i < 10; ++i)
    {
        std::cout << i << std::endl;
    }
}
```

# for文の文法図解

```
#include <iostream>
int main()
{ 反復
    for (int i = 0; i < 10; ++i)
    {
        std::cout << i << std::endl;
    } 繰り返し実行されるブロック
}
```

# for文の文法図解

```
#include <iostream>
int main()
{
    反復の初期設定式
    for (int i = 0; i < 10; ++i)
    {
        std::cout << i << std::endl;
    }
}
```

# for文の文法図解

```
#include <iostream>
int main()
{
    for (int i = 0; i < 10; ++i)
    {
        std::cout << i << std::endl;
    }
}
```

反復の継続条件式

# for文の文法図解

```
#include <iostream>
```

```
int main()
```

```
{
```

```
    for (int i = 0; i < 10; ++i)
```

```
    {
```

```
        std::cout << i << std::endl;
```

```
    }
```

```
}
```

次の反復に  
移る際に  
実行される式



# while 文

```
#include <iostream>
int main() {
    int x = 0;
    while (x >= 0)
    {
        std::cin >> x;
        std::cout << x << std::endl;
    }
}
```

青字の条件式を満たす限り、緑字のブロックを繰り返し実行

# 「n回だけ繰り返す」をwhile文で

```
#include <iostream>
int main() {
    int x = 0;
    while (x < 10)
    {
        std::cout << x << std::endl;
        x = x + 1;
    }
}
```

青字の条件式を満たす限り、緑字のブロックを繰り返し実行

# while 文から for 文への書き換え

```
#include <iostream>
```

```
int main() {  
    int x = 0;  
    for ( ; x < 10; )  
    {  
        std::cout << x << std::endl;  
        x = x + 1;  
    }  
}
```

空欄にすると何も実行しない

# while 文から for 文への書き換え

```
#include <iostream>
int main() {
    int x = 0;
    for (; x < 10; )
    {
        std::cout << x << std::endl;
        x = x + 1;
    }
}
```

反復の前に行う処理＝初期化式

次の反復に移る直前に行う処理

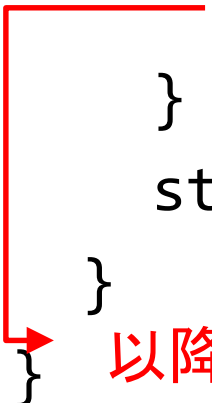
# 「n回だけ繰り返す」をfor文で

```
#include <iostream>
int main() {
    int x = 0;
    for (int x = 0; x < 10; x = x + 1)
    {
        std::cout << x << std::endl;
        x = x + 1;
    }
}
```

# 反復の中断: break

for, while 共通

```
#include <iostream>
int main() {
    while (true) {
        std::cin >> x;
        if (x < 0) {
            break;
        }
        std::cout << x << std::endl;
    }
}
```



以降の反復を中断して, while/for ブロックから抜け出す

# ブロック内処理の後略: continue

for, while 共通

```
#include <iostream>
int main() {
    for (int i = 0; i < 10; i = i + 1) {
        std::cin >> x;
        if (x < 0) {
            continue;
        }
        std::cout << x << std::endl;
    }
}
```

while/forブロック内の残りの文をスキップして、次の反復へ移る

# ブロックとコード整形

```
#include <iostream>
int main(void)
{
    int x = 0;
    while (x < 10)
    {
        std::cout << x << std::endl;
        x = x + 1;
    }
}
```

## ブロック:

- ・中括弧 {} で囲んだ領域
- ・処理のひとまとまり

インデントを揃えて**整列**  
ひとまとまりの処理は**近接**  
全体を通して一貫させる

mainの  
ブロック

whileの  
ブロック



# 変数xの範囲（使える範囲）

```
#include <iostream>
```

```
int main(void) {
```

```
    int x = 0;
```

```
    while (x < 10) {
```

```
        int y = x + 1;
```

```
        std::cout << y << std::endl;
```

```
        x = x + 1;
```

```
    }
```

```
    std::cout << y << std::endl; //エラー
```

```
}
```

xの宣言後 &&

xを宣言したブロック内

（その内側のブロックも含む）

# 変数 $y$ のスコープ(使える範囲)

```
#include <iostream>
```

```
int main(void) {
```

```
    int  $x$  = 0;
```

```
    while ( $x$  < 10) {
```

```
        int  $y$  =  $x$  + 1;
```

```
        std::cout <<  $y$  << std::endl;
```

```
         $x$  =  $x$  + 1;
```

```
    }
```

```
    std::cout <<  $y$  << std::endl; //エラー
```

```
}
```

$y$ の宣言後 &&  
 $y$ を宣言したブロック内  
(その内側のブロックも含む)

# 多重ループ

```
#include <iostream>
int main(void)
{
    for (int x = 0; x < 5; x = x + 1)
    {
        for (int y = 0; y < 5; y = y + 1)
        {
            std::cout << x + y << std::endl;
        }
        std::cout << x << std::endl;
    }
}
```

xが0の状態で yが0～4まで反復し、  
次にxが1の状態でyが0～4まで(略)  
次にxが2の(略)(略)  
最後にxが4の状態でyが(略)

# 変数xのスコープ

```
#include <iostream>
int main(void) {
```

for()も含むブロック内  
(その内側のブロックも含む)

```
    for (int x = 0; x < 10; x = x + 1) {
        for (int y = 0; y < 10; y = y + 1) {
            std::cout << x + y << std::endl;
        }
        std::cout << x << std::endl;
    }
```

```
}
```

# 変数 $y$ のスコープ

```
#include <iostream>
int main(void) {
    for (int  $x$  = 0;  $x$  < 10;  $x$  =  $x$  + 1) {
        for (int  $y$  = 0;  $y$  < 10;  $y$  =  $y$  + 1) {
            std::cout <<  $x$  +  $y$  << std::endl;
        }
        std::cout <<  $x$  << std::endl;
    }
}
```

for()も含むブロック内

# 変数宣言の重複

```
#include <iostream>
```

```
int main(void) {
```

青sumと赤sumは別物

```
    int sum = 0;
```

```
    for (int i = 0; i < 10; i = i + 1) {
```

```
        int sum = 0;
```

赤sumのスコープが優先

```
        sum = sum + i;
```

```
    }
```

```
    std::cout << sum << std::endl;
```

```
}
```

# 演算子の書き換え

## 算術演算＋代入

```
a = a + 5;
```

```
a = a - 2;
```

```
a = 6 * a;
```

```
a = a / 4;
```

## インクリメント

```
a = a + 1; a += 1;
```

## デクリメント

```
a = a - 1; a -= 1;
```

## 複合代入

```
a += 5;
```

```
a -= 2;
```

```
a *= 6;
```

```
a /= 4;
```

## インクリメント

```
++a; か a++;
```

## デクリメント

```
--a; か a--;
```

# 配列と反復：配列各要素の処理

```
#include <iostream>
int main(void) {
    double a[10], b[10];
    for (int i = 0; i < 10; ++i)
    {
        a[i] = i + 0.5;
        b[i] = a[i] * 2.0;
        std::cout << b[i] << std::endl;
    }
}
```



# 配列と反復：合計値

```
#include <iostream>
int main(void)
{
    int a[5] = {1, 2, 5, 8, 14};
    int sum = 0;
    for (int i = 0; i < 5; ++i) {
        sum += a[i];
    }
    std::cout << sum << std::endl;
}
```

# 配列と反復：フィボナッチ数列

```
#include <iostream>
int main(void) {
    int fib[10];
    fib[0] = 1;
    fib[1] = 1;
    for (int i = 2; i < 10; ++i) {
        fib[i] = fib[i - 1] + fib[i - 2];
        std::cout << fib[i] << std::endl;
    }
}
```

# まとめ

- for (初期化式; 継続条件式; 更新式) { }
- while (継続条件式) { }
- 反復の中断には break
- 反復対象ブロック内の後略には continue
- 変数のスコープは宣言した中括弧 { } 内
- スコープに対応したインデントで整形を
  - ついでに単語・演算子の前後の空白も

# 講義しなかったこと

- do~while 構文
- インクリメント ++i と i++ の違い
- デクリメント --i と i-- の違い

# 演習課題01: スケーリング

提出期限: 11/5(月)、ファイル名: 05-01.cpp

- ユーザー入力された5つの浮動小数を, それぞれ2倍して出力するプログラムを作成
  - 「Programming Basics」のページより, 「05-01.cpp」を取得し, 必要なコードを追加する
  - 値が入力される度に2倍した値を出力してもOK, あるいは5つの値を全て受け取った後に, それぞれ2倍した値を5つまとめて出力しても, どちらでもOK

# 演習課題02：特定要素の出力

提出期限：11/5(月)、ファイル名：05-02.cpp

- 配列に格納されている20個の数値のうち、その約数が2あるいは3である数値を全て出力するプログラムを完成
  - 「Programming Basics」のページより、「05-02.cpp」を取得し、必要なコードを追加する
  - 配列には整数 1～20 を格納
  - if を用いて実現 (continue も使うかも?)
  - 2か3で割り切れる数 = 余り (%) がゼロとなる数

# 演習課題Extra: ソート

提出期限: 11/5(月)、ファイル名: 05-ex.cpp

- 配列に格納されている20個の数値を, 小さい順に並べ替えるプログラムを作成
  - 「Programming Basics」のページより, 「05-ex.cpp」を取得し, 必要なコードを追加する
  - 配列にはランダムな整数を格納
  - ノーマルヒント「バブルソート」
  - アドバンスヒント「STLのsort」
  - アドバンスヒント「C標準ライブラリqsort」