プログラマー勉強会

作業ディレクトリの作成

Terminalを起動
↓
cd Desktop　を入力(デスクトップへ移動)
↓
mkdir pandd_prog　を入力
↓
cd pandd_progを入力
↓
mkdir ex01 ex02 ex03　を入力

※勉強会が終わったら、多分邪魔になるので消してもらって結構です。

プログラム実行までの手順

.cファイルにプログラムを書く
↓
コンパイルする
↓
実行する

コンパイル・・・書かれたプログラミング言語を機械語へ翻訳すること。

機械語・・・2進数(0と1だけの数)で表された言語。

実行・・・書いたプログラムを動かすこと。

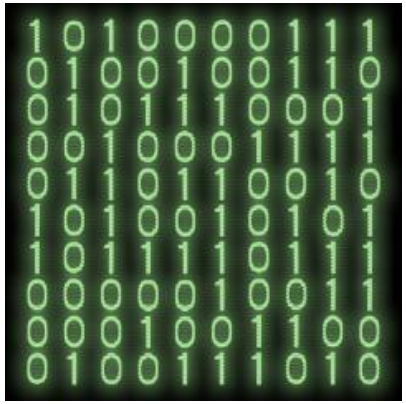
```c
#include <stdio.h>

void main(){
	int n,i;

	for(n = 1;n <= 100;n++){
		for(i = 1;i <= n;i++){
			if(i != 1&&n % i == 0){
				if(i == n){
					printf("%2d ",n);
				}else{
					break;
				}
			}
		}
	}
	printf("¥n");

	return 0;
}
```

コンパイル



プログラムを書いて実行してみよう

Terminalを起動
↓
cd ex01　を入力(ex01へ移動)
↓
emacs Hello.c &を入力 (プログラムを書くファイルを作成)

[Hello.c]

```
#include <stdio.h>

int main(){
    printf("Hello World¥n");
    return 0;
}
```

gcc Hello.c　を入力(コンパイル)
↓
./a.out　を入力(実行)
↓
Hello Worldと出ればOK

#includeとヘッダーファイル

#include <stdio.h>・・・プログラム中にstdio.hの中に入っている機能を使う。

.cファイルに書かれたプログラムだけでは、せいぜいできるのは計算くらい。
画面に文字列を表示する、ファイルを読み込む、その他複雑な処理をするには外部から機能を取り込む必要がある。

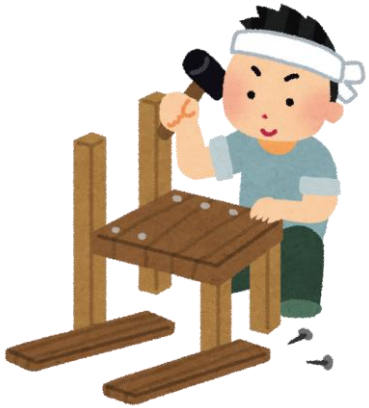
#include <ヘッダーファイル名>・・・これから書くプログラムのために、指定したヘッダーファイルを取り込む処理。

ヘッダーファイル・・・複雑な処理の機能が実装されたファイル。ライブラリなどと言う

#include

ヘッダーファイル

機能



処理できる

何だこれらは

int main(){}・・・メイン関数。プログラムはこの中に記述された命令文を上から順に実行していく。

printf();・・・stdio.hの中に入っている機能の一つ。()の中に書かれた文字列を画面に出力する役割を持つ。

＼n or ¥n・・・改行文字。書くと改行される。基本的に文末に置く。

return 0;・・・メイン関数を終了させる　≒　プログラムを終了させる役割を持つ。

gcc ファイル名.c・・・コンパイルコマンド。指定した.cファイルをコンパイルする。

./a.out・・・コンパイルしたプログラムを実行するコマンド。

今までのを参考に

ex1-1.cというファイルを作って、そこに

自分の学籍番号 自分のHN

を出力するプログラムを作ってください

解答

```
#include <stdio.h>

int main(){
    printf(“s1240180　アイトレ¥n”);
    return 0;
}
```

変数

emacs variable.c &を入力

[variable.c]

```
#include <stdio.h>

int main(){
    int n;
    n = 3;
    printf(“n = %d¥n”,n);
    return 0;
}
```

gcc variable.c　を入力
↓
./a.out　を入力
↓
n = 3と出ればOK

変数の宣言

変数・・・数値を保存する、入れ物的なやつ

変数を使うには、メイン関数の最初に変数を宣言する必要がある

[変数宣言のテンプレ]
変数の型名 変数名;

変数の型・・・保存したい数値の種類。整数型、小数点型など

[主な型名]
int・・・整数型
double・・・実数型(小数点型)

他にもcharとかfloatとかあるけど今回は省略

変数の宣言②

[変数名のルール]
なんでもいいというわけではない。

・頭文字・・・アルファベットと_(アンダーバー)
・2文字目以降・・・アルファベット、数字、_(アンダーバー)
・すでに宣言されている変数名は使ってはいけない

[OK]

_a、abc、s1、

[ダメ]

1a、-gc、g2,-

変数への代入

代入・・・変数へ数値を入れること。

[代入のやり方]
代入演算子　=　を使う。
数学では、演算子を挟んで左と右は等しいという意味だが、
プログラミングでは、
「演算子挟んで右の値を左の変数に代入する」という意味になる。

[例]
int n;　←int型の変数nを宣言
n = 3;　←nに3を代入

[宣言と同時に代入]
int n = 3;
このように変数の宣言と同時に代入もできる。これを変数の**初期化**と呼ぶ

代入には基本的に同じ型(種類)の数値を使う。
違う型でも代入できることがあるが、値が壊れることがある。

変数の表示

printfを使って、変数に保存された数値を表示する。
C言語では変数を表示するには出力変換指定子を使う。

出力変換指定子・・・変数に保存された数値を文字列に変換する記号。

[主な出力変換指定子]
%d・・・整数
%f・・・実数

[使い方]
printf内の""で囲まれた部分の数値を表示させたい場所に置き、""の後ろに,変数名と書く。

[例]
int型の変数nを表示させる。
printf("n = %d¥n",n); → n = 値と表示される。

変数の表示②

[複数の変数を表示させる]
変数を,で区切ってprintfの後ろに書き、変数の数だけ指定子を""内に置く。

printf(“%d %d %d¥n”,a,b,c);　前から順に1対1で対応する。

[例]

```
int a,b;
double c;
a = 1;
b = -3;
c = 2.5;
printf(“a = %d b = %d c = %f¥n”,a,b,c);
```

↓

a = 1 b = -3 c = 2.50000　と表示される

変数の注意点

①変数を表示するときは必ず値を代入してから表示する。
→宣言した段階では、変数の値は定まっていない(不定)
不定の状態で表示させるとでかい数値が表示される。

②int型に小数を入れると、
→小数点以下が値にかかわらず、すべて切り捨てられる。
[例]
int n = 3.5;　→　n = 3
3はもちろん、3.1も3.7も3.99999999999もすべて3になってしまう。

計算させてみる

変数に保存された数値や、ただの数値を四則演算と剰余算させてみる。

[やり方]
その演算に応じた演算子を使う。

[演算子]
＋　･･･足し算の演算子。
-　･･･引き算の演算子。
*　･･･掛け算の演算子。
/　･･･割り算の演算子。
%　･･･剰余算の演算子。

[例]
int型の変数nがあるとすると。

```
n + 3;
n – 6;
n * n;
n / 2;
n % 4;
```

計算結果の表示と保存

計算結果も数値なので、変数の時と同様に表示や保存ができる。

[表示]

```
int n;
n = 5;
printf(“n + 3 = %d¥n”,n+3);
```

→ n + 3 = 8 と表示される。

[変数へ保存]

```
int n,m;
n = 5;
m = n + 4;
printf(“n + 4 = %d¥n”,m);
```

→ n + 4 = 9 と表示される。

[double型とint型の演算]

double型とint型を演算すると演算結果の数値はdouble型となる。
そのため、表示には%fを使い、保存にはdouble型の変数を使う。

今までのを参考にして

ex1-2.c というファイルを作り、そこに以下の計算結果を表示するプログラムを作成してください。(n,fには好きな数を代入してください。)

```
int n;
double f;

n + 3
f + 100
n – 12
2.8 - f
n * 5
f * n
n / 3
f / 0.8
n % 7
```

解答

```
#include <stdio.h>

int main(){
    int n = 6;
    double f = 2.6;

    printf(“n + 3 = %d¥n”,n + 3);
    printf(“f + 100 = %f¥n”,f + 100);
    printf(“n – 12 = %d¥n”,n – 12);
    printf(“2.8 – f = %f¥n”,2.8 – f);
    printf(“n * 5 = %d¥n”,n * 5);
    printf(“f * n = %f¥n”,f * n);
    printf(“n / 3 = %d¥n”,
    printf(“f / 0.8 = %f¥n”,f / 0.8);
    printf(“n % 7 = %d¥n”,n % 7);
    return 0;
}
```

演算と代入を同時に行う

今までやってきた代入演算子と四則演算＋剰余算を組み合わせた
演算子が存在する。

int n = 3　と宣言されているとする。

+=　・・・　変数に値を加える

n += 2;　→　nは5になる。

-=　・・・　変数から値を引く

n -= 1;　→　nは2になる。

***=　・・・　変数に値をかける**

n *= 2;　→　nは6になる。

/=　・・・　変数を数値で割る

n /= 3;　→　nは1になる。

%=　・・・　変数を数値で割った時の余りを入れる

n %= 2;　→　nは1になる。

n = n ○ 数値;
と同じ意味。

インクリメントデクリメント

インクリメント・・・変数の値を１増やす処理。
デクリメント・・・変数の値を１減らす処理。

[使い方]

・インクリメント	・デクリメント
変数名++;(後置)	変数名--;(後置)
++変数名;(前置)	--変数名;(前置)

[例]

int n = 3;
n++;　→　nが１増えて4になる。
n--;　→　nが１減って3になる。

[前置と後置の違い]

前置・・・同じ行内で、増減させてから他の演算処理をする
後置・・・同じ行内の一通りの演算処理が終わってから増減させる

n = m++;　→　mをnに代入してからmを一つ増やす。
a = --b + c;　→　bを一つ減らし、cとの足し算を行った結果をaに代入

演算子の注意点

①/　割り算の注意点。
→ n / 0　・・・　のように割る数を0にしない。
　分数で表すと分母が0の状態になり、壊れた数値が生まれる
→ n / m　・・・のように割る数が変数のとき0になっていないか確認する
→ int型同士で演算するとき、割り切れる数でないと正確な値が出ない
　5 / 2　→　2.5になるはずだが、int型なので2になる。

②％剰余算の注意点
→ 演算するときはint型同士でなければならない。
　3 ％ 2.5　・・・　エラーになる。

値を入力してプログラムを動かす

emacs input.c &　と入力

[input.c]

```
#include <stdio.h>

int main(){
    int n;
    printf(“値を入力してください：”);
    scanf(“%d”,&n);
    printf(“入力された数値は%d¥n”,n);
    return 0;
}
```

実行すると

input.cをコンパイルして実行すると、値を入力してください：と表示され、
プログラムが止まる。
ここで、好きな数を入力して、Enterキーを押すと。
入力された数値は～　と表示される。

入力ツール

scanf()・・・stdio.hの中に入っているツールの一つ。
Terminalから入力された数値をプログラム中の変数に代入する。

[使い方]

scanf(“代入する変数に対応した指定子”,&変数名,…);

[例]

int型の変数nに代入したい。

→ scanf(“%d”,&n);

int型の変数a,b,cに代入したい。

→ scanf(“%d%d%d”,&a,&b,&c);

※複数の値を入力するときは、Terminal上で、半角スペースで区切って入力

double型の入力

printf()では、double型の変数の表示に%fの指定子を使用していたが、
scanf()では、double型の変数の入力には**%lf**を使う。

[例]
double型の変数fに代入したい。
→　scanf(“%lf”,&f);

double型の変数p,q,rに代入したい。
→　scanf(“%lf%lf%lf”,&p,&q,&r);

int型の変数nとdouble型の変数fに代入したい。
→　scanf(“%d%lf”,&n,&f);

今までのを参考にして

ex1-2.cを、scanfを使って変数n,fに代入し、計算結果を表示するプログラムに改造してください。

解答

```
#include <stdio.h>

int main(){
    int n;
    double f;

    printf(“値を入れてください：”);
    scanf(“%d%lf”,&n,&f);
    printf(“n + 3 = %d¥n”,n + 3);
    printf(“f + 100 = %f¥n”,f + 100);
    printf(“n – 12 = %d¥n”,n – 12);
    printf(“2.8 – f = %f¥n”,2.8 – f);
    printf(“n * 5 = %d¥n”,n * 5);
    printf(“f * n = %f¥n”,f * n);
    printf(“n / 3 = %d¥n”,
    printf(“f / 0.8 = %f¥n”,f / 0.8);
    printf(“n % 7 = %d¥n”,n % 7);
    return 0;
}
```

変数の値はどうなってる？

プログラム中演算が何度も行われることがある。
値が分からなくならないようにプログラムを追っかけよう。

```
#include <stdio.h>

int main(){
    int x = 0,y = 2,z = -1;

    x = y;
    y++;
    y = z--;

    z += 10;
    z *= ++x – y + 3;

    y %= 5 + x;
    x = y * (z - 5) + 7;

    printf(“x = %d y = %d z = %d¥n”,x,y,z);

    return 0;
}
```

おわり

出力～変数～演算～入力