1.4 C言語の文法

式と文

- 式とは : 値を持つもの
  - 1+(2*3) は 7 という値をもつ
  - 2 > 3 は 偽 という値をもつ
- 文とは : コンピュータへの命令
  - printf(“%g¥n”,f); は変数fの表示
- 式x=1+2+3によって
  - この式全体の結果は6
  - 変数xに値6が格納される
- 複文とは : 文の並び
  - {文; 文; …}

制御構造

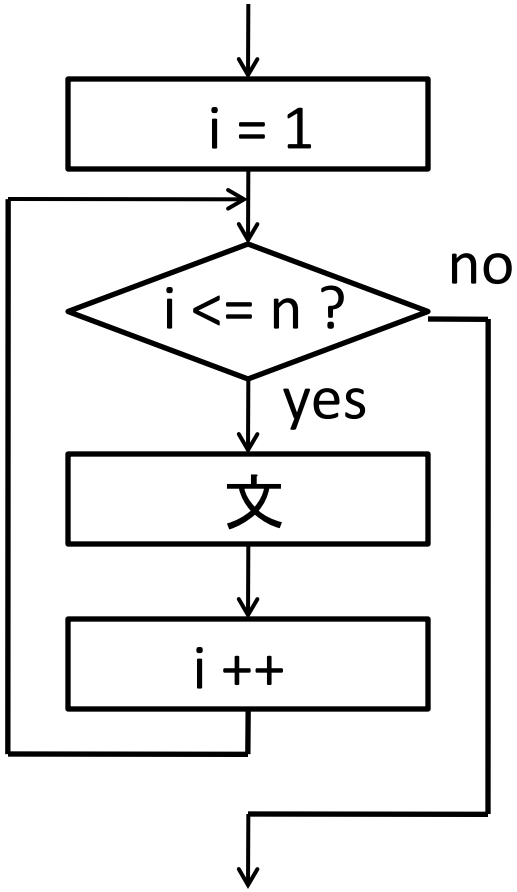
- 文を実行する順序を表現
  - 繰り返し(ループ)(for, while, do-while)
  - 条件分岐(if, switch)
  - ループからの脱出(break, continue)
  - ジャンプ(goto)

for文

- 繰り返し回数が決まっている
- for(i = 1; i <= n; i++) 文

  初期化　継続条件　更新

  - まずiに1を代入
  - i<=nが成立する間、繰り返し
  - 毎回iを1増やす

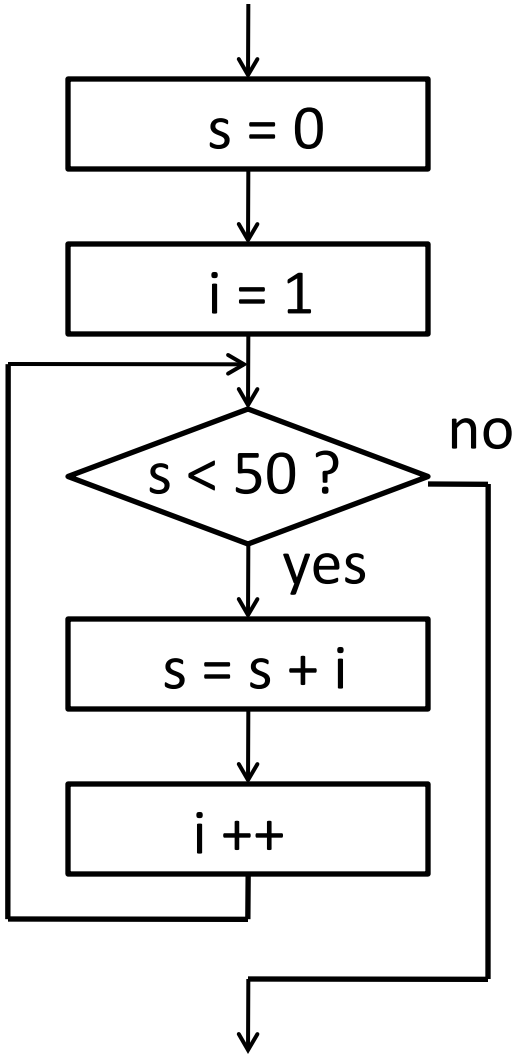


while文

- 繰り返し回数が決まっていない
- while(s < 50)文

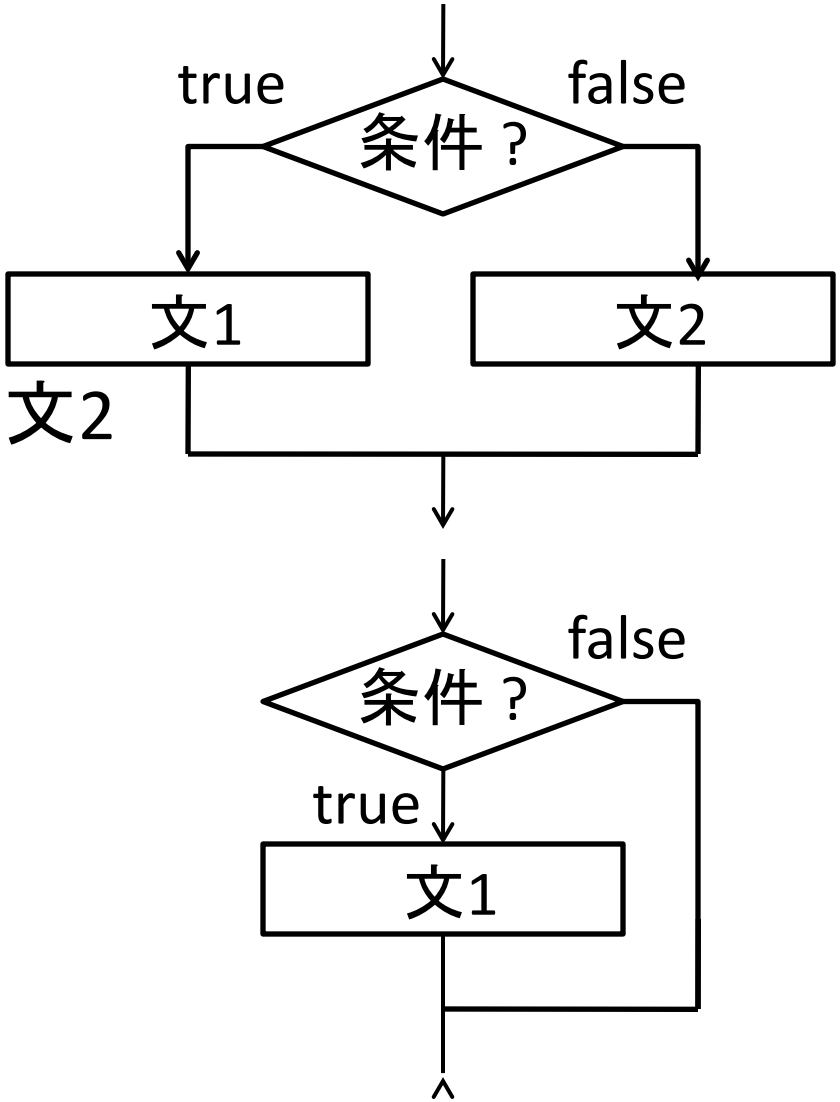
  条件

  – 条件をチェックし、成立なら文実行



if文

- 条件分岐
- if (条件) 文1 else 文2
  - 条件が真なら文1、偽なら文2を実行
- if (条件) 文1
  - 条件が真なら文1を実行



左右逆転した2進数を表示

```
#include <stdio.h>
int main(void) {
    int n;
    scanf("%d", &n);
    while (n > 0) {
        printf("%d", n%2);
        n = n / 2;
    }
    printf("¥n");
}
```

nを2で割った余り

nを2で割った商

nに6が入力されたとする

n%2 →0　　n/2 →3

n%2 →1　　n/2 →1

n%2 →1　　n/2 →0

011と表示される

最大公約数を求める

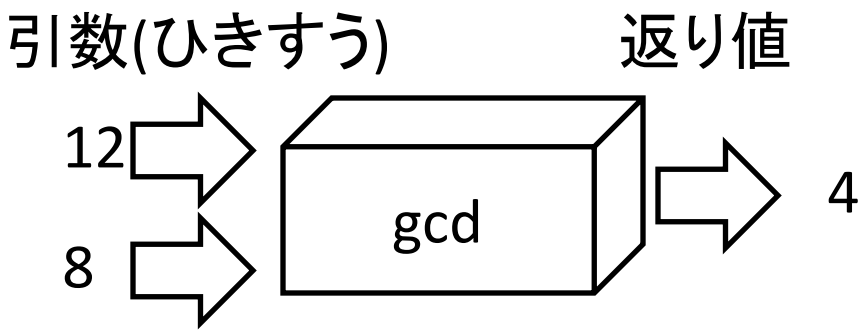
- ユークリッドの互除法
  - a, b (a ≧ b) について、a の b による剰余を r とすると、a と b との最大公約数は b と r との最大公約数に等しい
  - gcd(300, 24) = gcd(24, 12) = 12
  - greatest common divisor

関数定義

```
#include <stdio.h>
int gcd(int x, int y) {
    int r;
    r = x % y;
    while (r > 0) {
        x = y;
        y = r;
        r = x % y;
    }
    return y;
}
int main(void){
    int s,t,u;
    scanf("%d", &s);
    scanf("%d", &t);
    scanf("%d", &u);
    printf("%d¥n", gcd(gcd(s,t),u));
}
```

12 8

gcd(12,8)

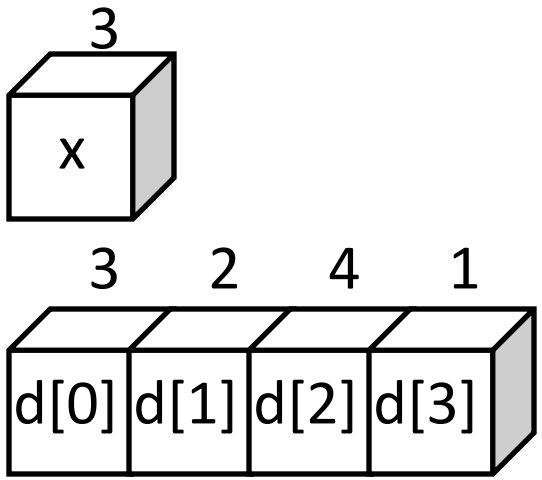


gcd(12,8)が呼ばれたとする

x=12, y=8, r=4 → r>0なのでwhile文実行

x=8, y=4, r=0 → r=0なのでyの値を返す

配列



- 変数 : 値が入る箱
- 配列 : 値が入る箱の並び
  - 同種の操作を行いやすい
- 例:10個の変数の値を全て0にする
  - 変数なら: x0=0; x1=0; x2=0; x3=0; x4=0; x5=0; x6=0; x7=0; x8=0; x9=0;
  - 配列なら: for (i = 0; i < 10; i++) d[i] = 0;

配列の書き方

- d[3]の3を添字と呼ぶ
- C言語では配列の添字は0から始まる
- プログラム中で
  - int d[31]; で配列宣言(d[0]からd[30]ができる)
  - 個々の要素はd[0], d[1], …, d[30]で参照できる

配列のプログラム例

```
#include <stdio.h>
#include <stdlib.h>
#define LENGTH 31
int main(void) {
    int n, i;
    int d[LENGTH];
    scanf("%d", &n);
    for (i = 0; i < LENGTH; i++)
        d[i] = 0;
    if (n<0)return EXIT_FAILURE;
```

入出力に必要なライブラリ

EXIT_…定数に必要なライブラリ

マクロ定義

配列の定義

10進数の入力

配列クリア

入力負なら終了

```
    for (i = 0; i < LENGTH; i++){
        if (n % 2 == 1) d[i] = 1;
        n = n / 2;
    }
    for (i = 0; i < LENGTH; i++)
        printf("%d", d[LENGTH-1-i]);
    printf("¥n");
    return EXIT_SUCCESS;
}
```

2で割った余りをd[i]に格納

配列を逆順に出力

改行

プログラムの説明

- 「左右反転した2進数を表示するプログラム」を思い出す

p.21

```
while (n > 0) {
    printf("%d", n % 2);
    n = n / 2;
}
```

2で割った余りを出力

p.24

```
for (i = 0; i < LENGTH; i++){
    if (n % 2 == 1) d[i] = 1;
    n = n / 2;
}
```

2で割った余りを配列に格納

- nとして13が入力されたとき
  - d[0] = 1, d[1] = 0, d[2] = 1, d[3] = 1, d[4] = … = 0
  - 逆順に表示すると00…001101

用語

- マクロ定義
  - プログラム中での具体的な数値(マジック・ナンバー)は避ける→LENGTHを使っている
    - 何の数字か後でわからなくなる
    - 何箇所も出てくると修正し忘れやすくなる
- main関数の返り値
  - main関数が正常に終了した時は0を返す
  - return 0; と書く代わりにreturn EXIT_SUCCESS;としている

1.5 コンピュータでの数

自然数、2進数、10進数、16進数

- 自然数: 一般社会では1からだが、コンピュータサイエンスでは0も自然数と考える

10進数	0	1	2	3	4	5	6	7	8	9	10
2進数	0	1	10	11	100	101	110	111	1000	1001	1010
16進数	0	1	2	3	4	5	6	7	8	9	a

10進数	11	12	13	14	15	16	17	18	…
2進数	1011	1100	1101	1110	1111	10000	10001	10010	…
16進数	b	c	d	e	f	10	11	12	…

10進数の定義

- 123は100のかたまりが1つ、10のかたまりが2つ、1のかたまりが3つある
  - 123 = 100×1 + 10 ×2 + 1 ×3

$$\sum_{j=0}^{n} 10^j d_j = (10^n \times d_n) + \ldots + (10^2 \times d_2) + (10 \times d_1) + d_0$$

- この式の10のところを2や16にすれば2進数、16進数になる

2進数と16進数

- 16進数の1桁は2進数の4桁に対応
- 2進数→16進数
  - 4桁毎に区切って変換
  - 1010|1011|1100 → abc
- 16進数→2進数
  - 1桁で2進数4桁を表す
  - d3e →1101|0011|1110
- 2進数と8進数も同様