2 ソフトウェア工学
Software Engineering

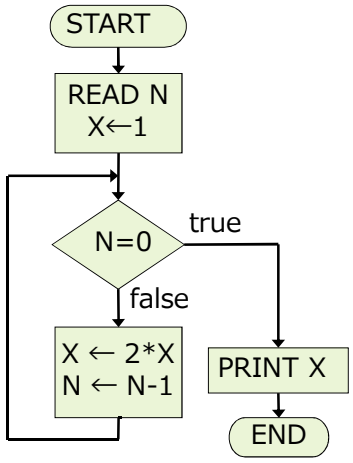
構造化プログラミング

STRUCTURED PROGRAMMING

動機(1/3) GO TO 文

GO TO 文：機械語の分岐命令に対応した制御文

■2 の N 乗を計算する流れ図 (flowchart)

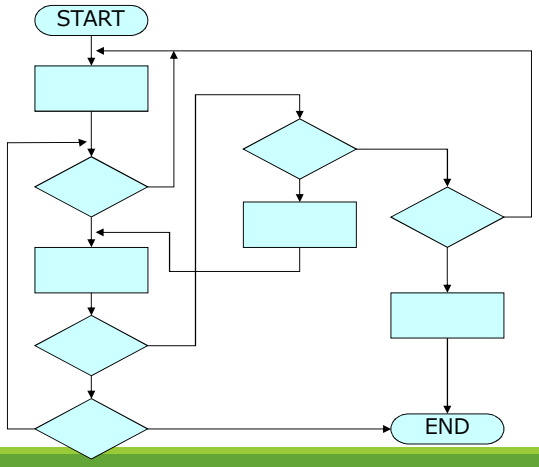


（初期の頃）
BASIC言語のプログラム

```
10 READ N
20 LET X=1
30 IF N=0 THEN GOTO 70
40 LET X=2*X
50 LET N=N-1
60 GOTO 30
70 PRINT X
80 END
```

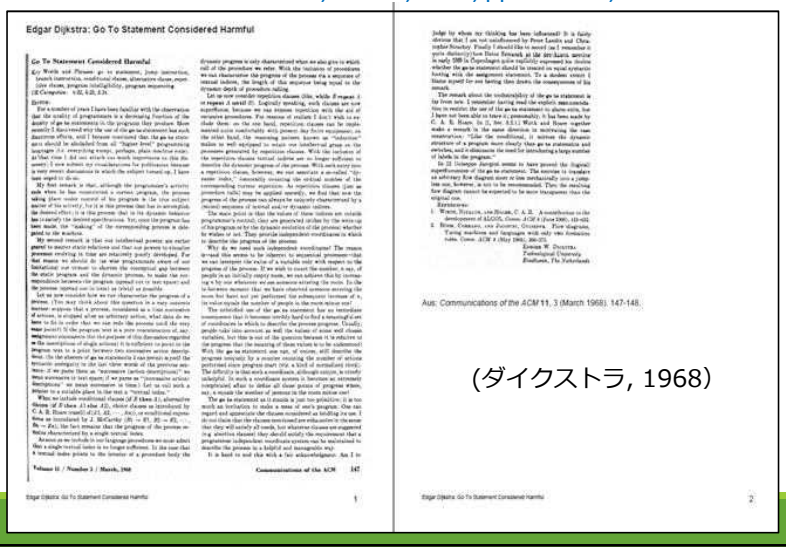
動機(2/3) スパゲティ・プログラム

スパゲティ・プログラム (spaghetii program)
GO TO 文を不規則に用い，流れ図の線が交差し，
プログラムの全体を把握しにくいプログラム



動機(3/3) GO TO 文有害説

E. Dijkstra: Go To Statement Considered Harmful,
Communications of the ACM, vol. 11, no. 3, pp.147-148, 1968.


Edgar Dijkstra: Go To Statement Considered Harmful

Go To Statement Considered Harmful

Aus: Communications of the ACM 11, 3 (March 1968), 147-148.

(ダイクストラ, 1968)

構造化プログラミング(1/6) 構造化定理

構造化定理

どんな流れ図も，３つの基本構造

- 順次 (sequence)
- 選択 (selection)
- 反復 (iteration)

の組合せにより，等価な

構造化流れ図 (structured flowchart)

に変換できる.

GO TO 文の使用制限

↓

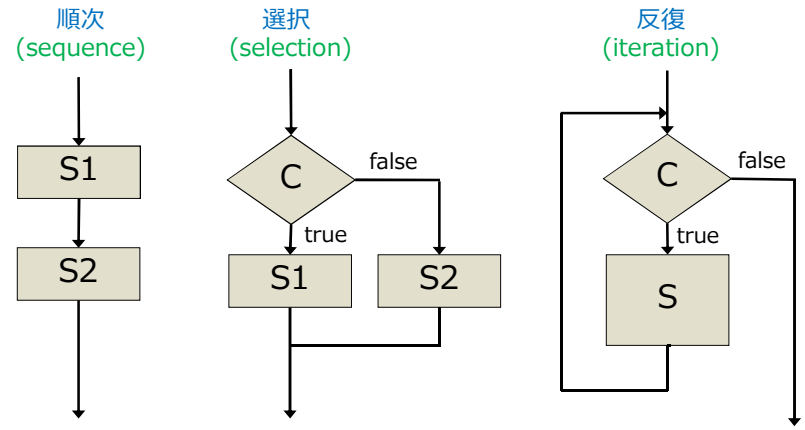
○ 流れ図の線の交差なし

○ 出入り口が１カ所

↓

構造化プログラミング
Structured programming

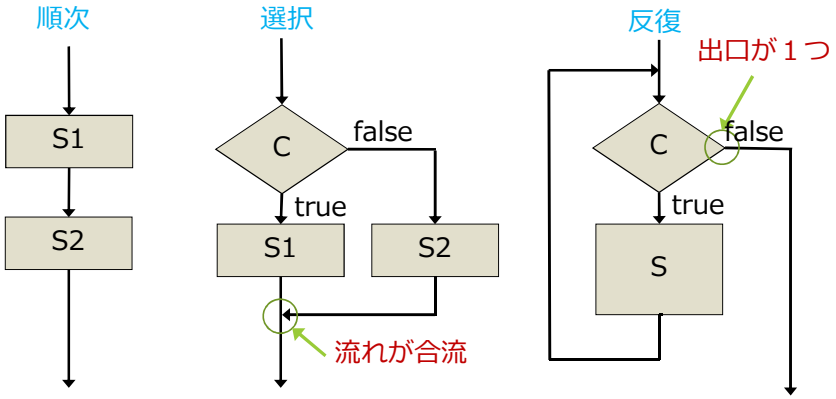
構造化プログラミング(2/6) ３つの基本構造



構造化プログラミング言語

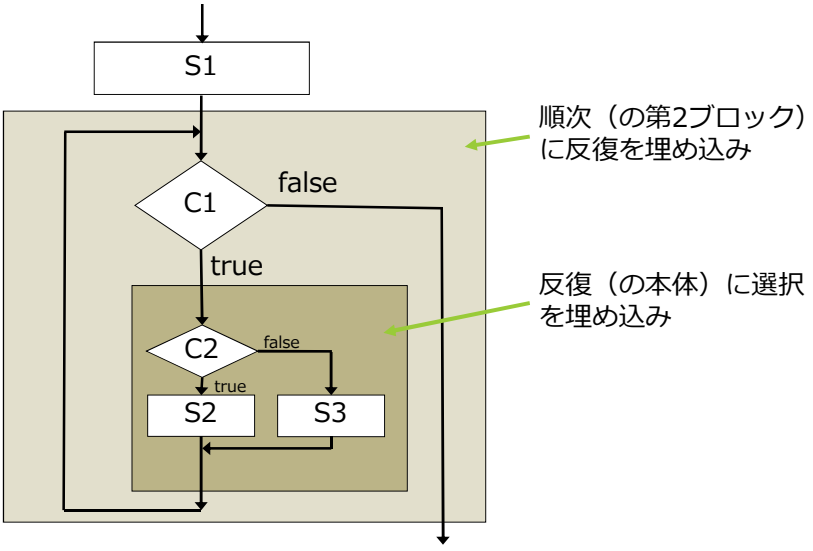
構造化プログラミング(3/6) 構造化流れ図

S, S1, S2自体に，基本構造を入れ子的に埋め込んだ流れ図

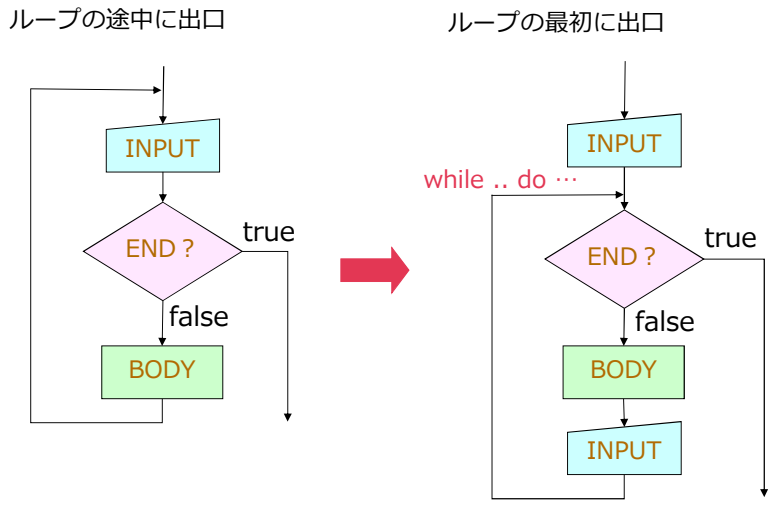


- 出入口はそれぞれ１カ所
- 流れを表す線が交差しない　（数学的に証明できる）

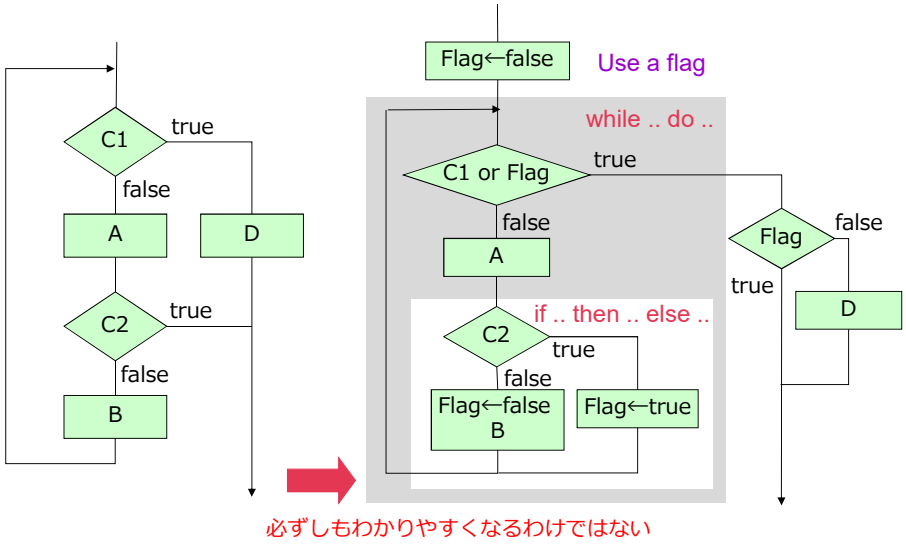
構造化プログラミング(4/6) 構造化流れ図の例



構造化プログラミング(5/6) 構造化流れ図への変換



構造化プログラミング(6/6) やや複雑な変換



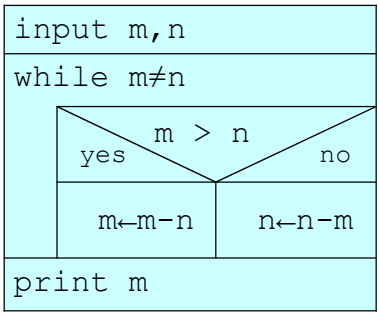
必ずしもわかりやすくなるわけではない

構造の視覚的表現(1/6) NSチャート

プログラムの構造をビジュアルにわかりやすくする

【例】最大公約数を求めるプログラム

NSチャート
(Nassi-Shneiderman diagram)



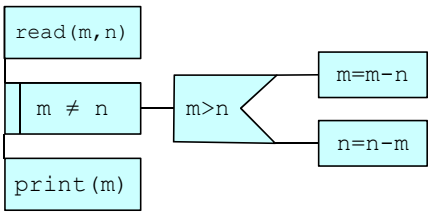
積み木のような表現

BASIC によるプログラム

```
input m,n
while m<>n
  if m>n then
    let m=m-n
  else
    let n=n-m
  end if
end while
print m
```

構造の視覚的表現(2/6) PAD

PAD (Problem Analysis Diagram)



木構造により表現
（日本人が考案し世界標準化）

Pascal によるプログラム

```
read(m,n);
while m<>n do
  if m>n then m=m-n
         else n=n-m;
print(m);
```

構造の視覚的表現(3/6) 字下げ

字下げ (Indentation)　細かな流儀がいろいろ。一貫して使うこと。

C 言語によるプログラム

良い例

```
int main(void){
  scanf(”%d %d”, &m, &n);
  while(m!=n){
    if(m>n){
      m=m-n;
    }
    else{
      n=n-m;
    }
  }
  printf(”%d¥n”, m);
}
```

構造の視覚的表現(4/6) 字下げの悪い例

C 言語によるプログラム

悪い例

```
int main(void){
  scanf(”%d %d”, &m, &n);
  while(m!=n){
    if(m>n){
      m=m-n;
  }
  else{
  n=n-m;
}
  }
  printf(”%d¥n”, m);
}
```

構造の視覚的表現(5/6) Python

字下げが意味を持つプログラミング言語もあるので注意！

Python によるプログラム

```
while m!=n:
  if m>n:
    m=m-n
  else:
    n=n-m
print m
```

```
while m!=n:
  if m>n:
    m=m-n
  else:
    n=n-m
    print m
```

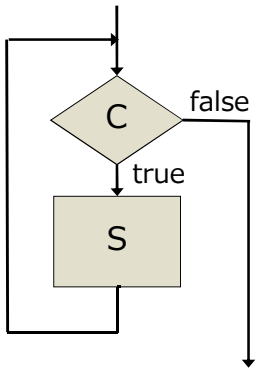
print m 実行のタイミング	while文の終了後	if文のelse部

構造の視覚的表現(6/6) Scratch

小学生などのプログラミング教育用
（GUIで積み木を組み立ててプログラムとする）

現実には他の構造も許す(1/3) 後判定反復

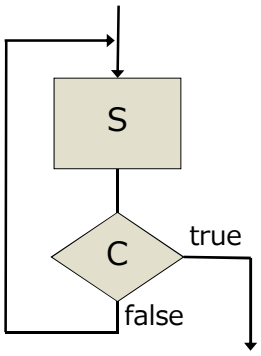
前判定反復



while C do S

S は0回以上実行される

後判定反復



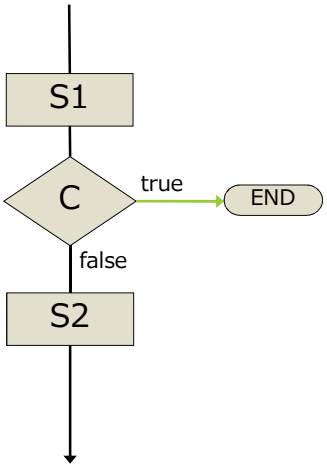
repeat S until C

あるいは do S while !C

S は1回以上実行される

現実には他の構造も許す(2/3) return文

return 文： 関数の出口に飛ぶ特別な GO TO 文

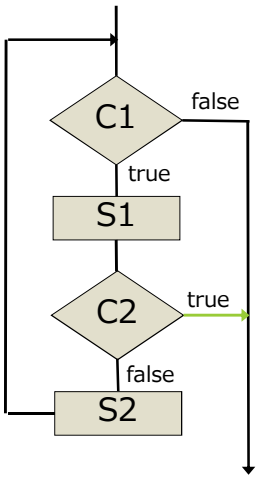



```
S1;
if C then return;
S2;
```

if 文の枝分かれが合流せず
出口が２カ所以上となるが
制御構造はあまり複雑化しない

現実には他の構造も許す(3/3) break文

break 文： ループの出口に飛ぶ特別な GO TO 文




```
while C1 do
    S1;
    if C2 then break;
    S2;
```

ループからの出口が 2 カ所以上になるが
制御構造はあまり複雑化しない

演習問題 2

つぎのCプログラムの構造を
(1) NSチャート
(2) ＰＡＤ
(3) 字下げ
により，それぞれ表示しなさい。

```
while((start < strlen(line))
&& !alpha(line[start]))
start++;
if(start >= strlen(line)) return -1;
else {printf("%c", line[start]);
start++;
while((start < strlen(line)) &&
(alpha(line[start])||num(line[start]))){
printf("%c", line[start]);
start++;}
printf("¥n");
return start;
}
```