定数と変数

定数

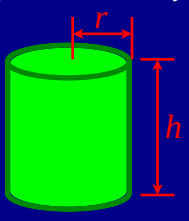
プログラム実行中に変化することのない「値」を保持するユーザ定義名前付オブジェクト

変数

プログラム実行中に変化する「値」を示すユーザ定義名前付オブジェクト

[例題]

円柱の半径と高さ読み込んで、その体積と表面積と表示するプログラムを作成せよ



体積 $V = \pi r^2 h$

表面積

$$S = 2\pi r^2 + 2\pi rh = 2\pi r(r + h)$$

定数

円周率πを参照する定数

定数の定義(Pascal)

const **定数名 = 定数値**;

program**文の後、変数宣言文の前**

複数の定数を定義する場合

const **定数名 = 定数値**;
定数名 = 定数値;
定数名 = 定数値;

定数の定義(FORTRAN)

PARAMETER(**定数名 = 定数値**)

program**文の後、変数宣言文の前**
簡単な演算も可能

複数の定数を定義する場合

PARAMETER(N = 10, M = 11)
PARAMETER(N = 10, M = N + 1)
PARAMETER(N = 10)
PARAMETER(M = N + 1)

定数の定義(c)

定数定義はマクロにより行う

#define **定数名 定数値**

#ifdef __小学生__
#define PI 3.14
#else
#define PI 3.141592
#endif

定数の定義

Pascal

```
const PI=3.141592;
```

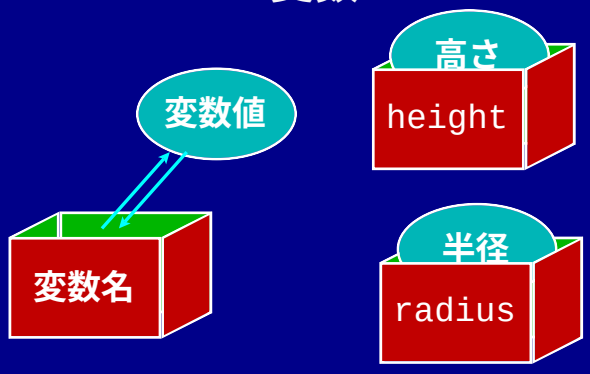
FORTRAN

```
PARAMETER (PI=3.141592)
```

c

```
#define PI 3.141592
```

変数



箱(変数)の在りか(アドレス)を入れる箱がポインタ変数

変数の種類

整数型

符号付き、符号なし(c)

実数型

倍精度(c, FORTRAN), 4倍精度(FORTRAN)

複素数型

二つの実数の組(FORTRAN)

文字型

文字符号化方法に依存する

整数型変数(定数)の表現

2進数のビット列(例8ビット)

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

$b_7\times2^7 + b_6\times2^6 + b_5\times2^5 + b_3\times2^3 + b_2\times2^2 + b_1\times2^1 + b_0\times2^0$

0	1	0	1	1	1	0	0

= 64+16+8+4 = 92

通常16ビットか32ビット

全てのビットを使用

$0 \sim 2^n - 1$ (符号なし整数)

負の整数型変数(定数)の表現

0	1	0	1	1	1	0	0

= 64+16+8+4 = 92

1の補数

1	0	1	0	0	0	1	1

= -92

0の定義が一意的でない(1111=0000)

2の補数

11111111 + 1 = 100000000 -1 + 1 = 0

1	0	1	0	0	1	0	0

= -92

減算を補数と加算で処理できる

整数型変数(定数)の表現

nビット符号なし整数

$0 \sim 2^n - 1$

nビット符号付整数(2の補数表現)

$-2^{n-1} \sim 2^{n-1} - 1$

実数の表現

固定小数点実数

24.5

浮動小数点実数

2.45×10^1, 2450×10^{-2}

正規化された浮動小数点実数

仮数部 0.245×10^2 **指数部**

2進固定小数点実数の表現

$b_3b_2b_1b_0 \cdot c_1c_2c_3$

$= b_3\times2^3 + b_2\times2^2 + b_1\times2^1 + b_0$

$+ c_1\times2^{-1} + c_2\times2^{-2} + c_3\times2^{-3}$

例

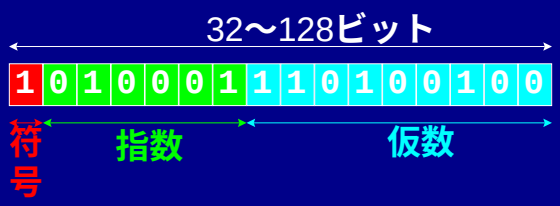
$101.011_2 = (4+1+1/4+1/8)_{10} = 5.375_{10}$

実数型変数(定数)の表現

先頭ビットで符号を表す

指数部は2進整数

仮数部は正規化された固定小数点



実数型変数(定数)の表現

仮想的8ビット実数

指数は10のべきを2の補数表現の整数

仮数は正規化された2進小数

1	0	1	0	1	1	0	1

$=-(0.5+0.25+0.0625)\times10^2=-81.25$

0	1	1	1	1	0	0	1

$=+(0.5+0.0625)\times10^{-1}=0.05625$

複素数型変数(定数)の表現

二つの実数で表す

変数の宣言(Pascal)

var 変数名: 変数の型;

定数定義の後、begin endの前

複数の変数を宣言する場合

var 変数名, 変数名, 変数名: 変数の型;
変数名: 変数の型;
変数名, 変数名: 変数の型;

使用できる変数の型(Pascal)

integer
- 符号付き整数
- 多くの処理系が16ビットか32ビット

real
- 単精度実数
- 多くの処理系は32ビット

例　var height, radius: real;
number: integer;

変数の宣言(FORTRAN)

変数の型 変数名

定数定義の後、最初の実行文の前

暗黙の宣言も可能
- A-H, O-Z から始まる変数名は実数型
- I, J, K, L, M, Nから始まる変数名は整数型

複数の変数を宣言する場合

変数の型 変数名, 変数名, 変数名

使用できる変数の型(FORTRAN)

INTEGER
- 符号付き整数、多くの処理系がか32ビット

REAL
- 単精度実数、多くの処理系は32ビット

DOUBLE PRECISION　(REAL*8)
- 倍精度実数、多くの処理系で64ビット

COMPLEX*16
- 多くの処理系で128ビット複素数

変数の宣言(c)

変数の型 変数名;

どこでも

宣言と同時に初期化も可能

変数の型 変数名 = 変数の初期値;

double radius = 0.25;

複数の変数を宣言する場合

変数の型 変数名, 変数名, 変数名;

使用できる変数の型(c)

int
- 符号付き整数、多くの処理系がか32ビット

float
- 単精度実数、多くの処理系は32ビット

double
- 倍精度実数、多くの処理系で64ビット

変数の宣言

Pascal

```
var r, h: real;
```

FORTRAN

```
REAL*8 HS, V
```

c

```
double v, s;
```

四則演算と代入 (Pascal)

和・差・積

変数名 := 変数(定数)名 + 変数(定数)名
変数名 := 変数(定数)名 – 変数(定数)名
変数名 := 変数(定数)名 * 変数(定数)名

商

変数名 := 変数(定数)名 / 変数(定数)名
変数名 := 変数(定数)名 div 変数(定数)名

四則演算と代入 (FORTRAN, c)

和・差・積・商

変数名 = 変数(定数)名 + 変数(定数)名
変数名 = 変数(定数)名 – 変数(定数)名
変数名 = 変数(定数)名 * 変数(定数)名
変数名 = 変数(定数)名 / 変数(定数)名

整数と実数の演算には型の変換を明示的に行うべし

例題(Pascal)

```
program cylind(input, output);
const pi = 3.141592;
var r, h, s, v: real;
begin
  read(r, h);
  writeln(r, h);
  v := pi*r*r*h;
  s := 2*pi*r*(r + h);
  writeln(v, s);
end.
```

例題(FORTRAN)

```
PROGRAM CYLIND
PARAMETER (PI = 3.141592)
DOUBLE PRECISION R, H, S, V
READ (5, *) R, H
WRITE (6, *) R, H
V = PI*R*R*H
S = 2.0*PI*R*(R + H)
WRITE (6, *) V, S
STOP
END
```

例題(c)

```
#include <stdio.h>
#define PI 3.141592
int main(void) {
  double r, h, s, v;
  scanf("%lf %lf", &r, &h);
  printf("%lf %lf\n", r, h);
  v = PI*r*r*h;
  s = 2.0*PI*r*(r + h);
  printf("%lf %lf\n", v, s);
  return 0;
}
```

Pascalの基本構造

program **プログラム名** (**入出力**);
const **定数名**=**定数**;
var **変数名**: **変数の型**;
begin
　実行すべき処理
end.

FORTRAN77の基本構造

PROGRAM **プログラム名**
PARAMETER(**定数名**=**定数定義**)
変数の型 変数名
実行すべき処理
STOP
END

c言語の基本構造

#プリプロセッサ命令
#定数マクロ定義
int main(**引き数**) {
　変数の型 変数名;
　実行すべき処理;
　return 0;
}

標準入力からの数値の入力

Pascal

read(**変数名**, **変数名**)

FORTRAN

READ (5, *) **変数名**, **変数名**

c言語

scanf("**入力書式**", &**変数名**, &**変数名**)

標準出力への数値の出力

Pascal

writeln(**変数名**, **変数名**)

FORTRAN

WRITE(6, **書式文番号**) **変数名**, **変数名**

c言語

printf("**出力書式**", **変数名**, **変数名**)