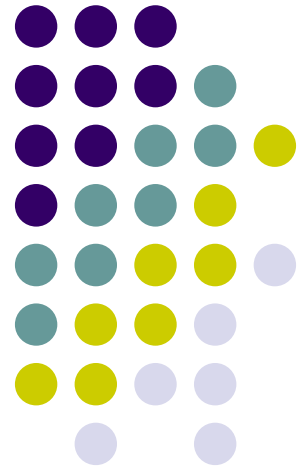
情報処理概論

工学部 物質科学工学科
応用化学コース
機能物質化学クラス

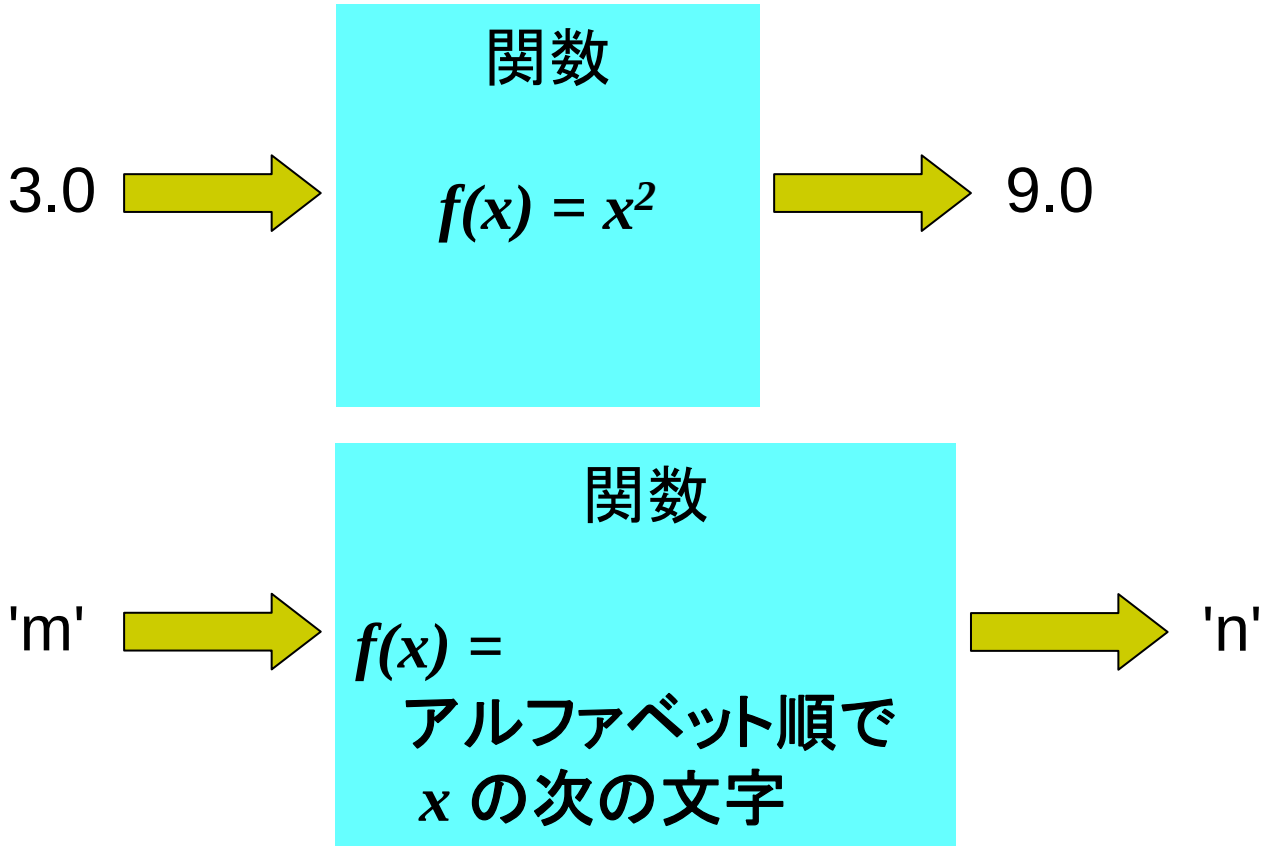


第９回

２００５年　６月１６日

関数（函数）とは

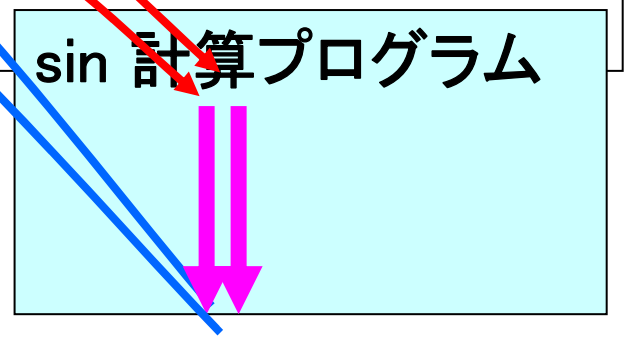
- 対応関係：　あるデータに対応する値を返す



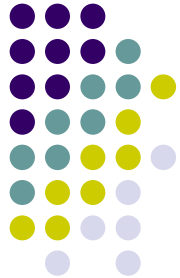
Fortranにおける関数

- 予め用意されているプログラムの呼び出し
  - 渡した値（引数）に応じた結果（返り値）を取得

```
program test4
  implicit none
  real(8) :: h, l, r
  intrinsic sin

  read(*, *) r

  l = r * sin(0.3333333D0)          ! 円に内接する三角形の一変の長さ
  h = l * sin(0.6666667D0)          ! 円に内接する正三角形の高さ

  write(*, *) 'answer: ', l * h / 2.0D0   ! 面積

stop
end program
```



自分で関数を定義する

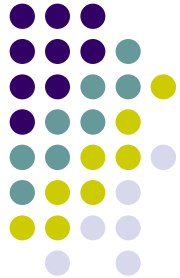


- 呼び出す関数の中身を自分で作成する.

- 場所： メインプログラムの外側
  - メインプログラムとは
    program から
    end program まで
  - メインプログラムの
    前、後どちらに置いても可

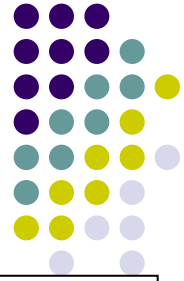
```
function test1(a)
 ...
end function


program triangle
 ...
end program


function test2(a)
 ...
end function
```

関数の例



- 例1） 第一回レポートで用いた近似法で
  $\sin x\pi$ の値を計算し、返す関数
- 例2） 2つの数のうち大きい方の値を返す関数
- 例3） 配列の平均値を返す関数

例1） 級数展開で sin xπを計算する関数

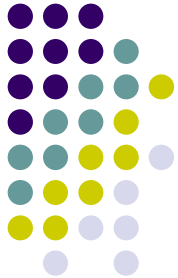


```
function mysin(x)
  implicit none
  real(8) :: x, y, mysin, rad, total
  integer :: i, h
  intrinsic :: atan, dble

  rad = 4.0D0 * atan(1.0D0) * x
  total = rad
  y = rad
  do i = 2, 100
    h = (2 * i - 2) * (2 * i - 1)
    y = -1.0D0*y*rad*rad / dble(h)
    total = total + y
  end do

  mysin = total
end function
```

関数mysinを呼び出すプログラム

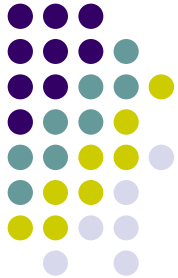
```
program test
  implicit none
  integer :: i
  real(8) :: k
  real(8),external :: mysin
  intrinsic dble
  do i = 1, 10
    k = dble(i) / 10.0D0
    write(*, '(a,f5.3,a,f13.10)') 'sin(', k, '*pi)= ',
mysin(k)
  end do
stop
end program

function mysin(x)
        前のページの通り
  ...

end function
```

例2） 2つの値のうち大きい方の値を返す関数



```
program test2
  implicit none
  real(8) :: a, b
  real(8), external :: maximum
  read(*, *) a, b
  write(*, *) maximum(a, b), ' is bigger.'
stop
end program

function maximum(x, y)
  implicit none
  real(8) :: x, y, maximum

  if (x > y) then
    maximum = x
  else
    maximum = y
  end if
end function
```

例3）配列の平均値を返す関数

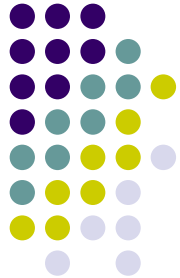


```
function average(n, a)
  implicit none
  integer :: n, i
  real(8), dimension(n) :: a
  real(8) :: average, total
  intrinsic dble

  total = 0.0D0
  do i = 1, n
    total = total + a(i)
  end do

  average = dble(total) / dble(n)

end function
```

平均値を返す関数を用いたプログラム例

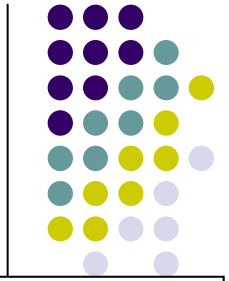
```
program test3
  implicit none
  integer, parameter :: n = 100
  integer :: i
  real(8), dimension(n) :: score
  real(8), external :: average

  open(10, file='score1')
  do i = 1, n
    read(10, *) score(i)
  end do
  write(*, *) 'Average = ', average(n, score)
stop
end program

function average(n, a)

  ...   前のページの通り

end function
```

関数を利用する利点



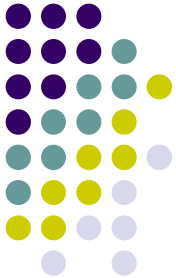
- 同じ処理を複数回実行するプログラムで、処理の記述を一箇所で済ませることができる.
  - プログラム内で何度も呼び出し可能
- 関数を抜き出して別のプログラムで利用可能
- プログラム全体の流れが見えやすくなる
  → 修正を行いやすい

関数の利用例



```
program test4
  implicit none
  real(8) :: h, l, r
  real(8),external :: mysin
  read(*, *) r

  l = r * mysin(0.3333333D0)        ! 円に内接する三角形の一変の長さ
  h = l * mysin(0.6666667D0)        ! 円に内接する正三角形の高さ

  write(*, *) 'answer: ', l * h / 2.0D0  ! 面積

stop
end program

function mysin(x)

  ...

end function
```

関数の利用法

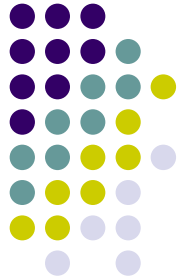


- 関数の定義

```
function 関数名(引数)
  implicit none
  変数,関数の宣言(引数やこの関数自身の宣言も行う)

  ... 計算 ...

  関数名 = 式
end function
```

- メインプログラムの外
- 関数自身も変数として宣言する
- 関数と同名の変数に格納された値が「返り値」として呼び出し側に返される.

関数定義の例



```
function average(n, a)
  implicit none
  integer :: n
  integer, dimension(n) :: a
  real(8) :: average
  integer :: i, total
  intrinsic dble

  total = 0
  do i = 1, n
    total = total + a(i)
  end do

  average = dble(total) / dble(n)

end function
```

引数の宣言

引数の宣言

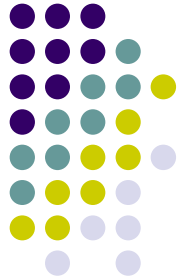
関数の宣言

関数内で用いる変数の宣言

関数内で用いる関数の宣言

返り値の計算，代入

関数の利用法



- 関数の宣言

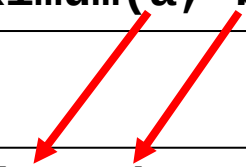
```
データ型名, external :: 関数名
```

- 関数の呼び出し
  - 通常の関数と同じ
  - 引数の順番に注意
  - 関数からさらに他の関数を呼び出しても良い
    - 呼び出し側の関数の中で，呼び出される側の関数の宣言が必要

関数の引数（ひきすう）

- 呼び出し側で指示された変数の値や数値が関数側の同じ位置にある引数にコピーされる
  - 名前ではなく順番が重要

```
  write(*, *) maximum(a, b), ' is bigger.'
```



```
function maximum(x, y)
  implicit none
  real(8) :: x, y, maximum
```

関数の利用に関する注意1

- 変数の有効範囲
  - 宣言されているメインプログラムや関数の中のみ

```
program test5
  implicit none
  real(8) :: x, y
  real(8), external :: sa
  x = 2.0D0
  y = 1.0D0
  write(*, *) 'answer = ', sa(y, x)
stop
end program

function sa(x, y)
  implicit none
  real(8) :: x, y, sa

  sa = x - y

end function
```

メインプログラムの x, y と
関数 sa 内の x, y は
別の変数なので、
このプログラムでは
　answer = -1.0000000
となる。
（1.0000000　ではない）

関数の利用に関する注意2

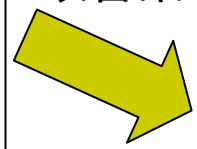
- 引数の値を変更すると誤動作する場合がある
  - なるべく、変数を使って値を渡す.

```
program test6
  implicit none
  real(8), external :: f

  write(*, *) f(100), 100
stop
end program

function f(n)
  implicit none
  integer :: n
  real(8) :: f
  f = n * n
  n = 0
end function
```

このプログラムの実行結果は
10000.00000000000 0
となる。

10000.00000000000 100
にならないことに注意.

改善策



```
...
real(8) :: t
real(8), external :: f

t = 100
write(*, *) f(t), 100
...
```

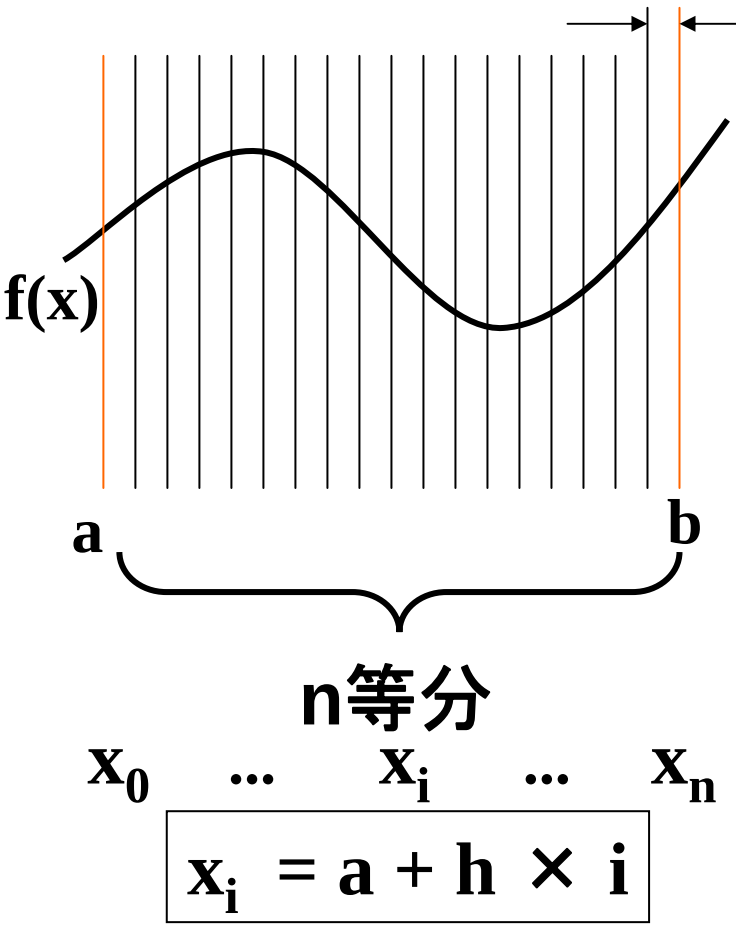
intrinsic と external

- intrinsic
  - Fortranの規格で定義されている組み込み関数
  - 講義の Webページから辿れる
    「Fortran90 プログラミング」
    の付録を参照

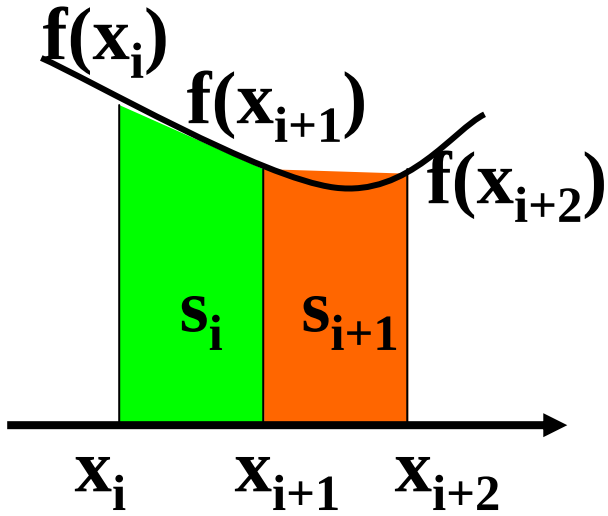
- external
  - プログラマが定義した関数

演習：台形積分のプログラム

- 台形積分



区間 $[x_i, x_{i+1})$ の積分を台形で近似



台形積分

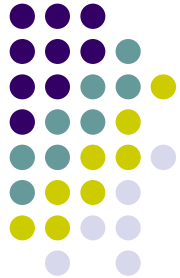
- 1区間の面積

$$s_i = (f(x_i)+f(x_{i+1})) \times h / 2$$

- 全区間の面積

$$\begin{aligned} S &= s_0 + s_1 + \ldots + s_{n-1} \\ &= (h / 2) \times (f(x_0)+f(x_1)+f(x_1)+f(x_2)+f(x_2)+f(x_3) \\ &\qquad \ldots +f(x_{n-1})+f(x_n)) \\ &= (h / 2) \times (f(x_0) + f(x_n) + 2 \times (f(x_1) + \ldots + f(x_{n-1})) \\ &= (h / 2) \times (f(a) + f(b) + 2 \times (f(x_1) + \ldots + f(x_{n-1})) \end{aligned}$$

穴埋め問題



- 次のプログラムの　　部分を埋めて
  台形積分プログラムを完成させる
- このプログラムは
  - 区間数が1000の場合の関数 f=sin(x) の
    台形積分の結果を表示する
  - 比較のため, 最後に関数　f を実際に積分した
    関数 ff を用いて定積分を計算し結果を表示する

```
% ./trapezoid
 Start point?
1.0
 End point?
2.0
Trapezoid:   0.956449062711
Integral:    0.956449142415
```

台形積分プログラム(1／2)

```
program trapezoid
  implicit none
  integer, parameter :: n = 1000
  integer :: i
  real(8) :: start, end, s1, s2 h
  [redacted]

  write(*, *) "Start point? "
  read(*, *) start
  write(*, *) "End point? "
  read(*, *) end

  h = (end - start) / dble(n)
  s1 = 0.0D0
  do i = 1, n-1
    s1 = s1 + [redacted]
  end do
  s1 = (s1 * 2.0D0 + [redacted] + [redacted]) * h / 2.0D0

  write(*, '(A, F15.12)') 'Trapezoid: ', s1
```

関数 f と ff の宣言

関数の呼び出し

関数の呼び出し

台形積分プログラム(2／2)

```
  s2 = [          ] - [          ]
  write(*, '(A, F15.12)') 'Integral:  ', s2
stop
end program

function f(x)
  implicit none
  [                    ]
  [                    ]
  intrinsic sin

  [                              ]
end function

function ff(x)
  implicit none
  [                    ]
  [                    ]
  intrinsic cos

  [                              ]
end function
```

関数の呼び出し

引数の宣言

関数の宣言

返り値の計算，代入

引数の宣言

関数の宣言

返り値の計算，代入

時間が余ったら

- 関数の中身を変えてみる

例）

$f(x) = x$

$ff(x) = x^2 / 2$

- n を変えて，精度の変化を確認する