

11042 計算機言語 6 回目

サポートページ:<https://goo.gl/678wGM>

石井 史之

金沢大学 数物科学系

November 20, 2017

計算機言語

- ガイダンス・・・ 10/2
- 復習 1(print 文, 繰返し等)・・・ 10/16
- 復習 2(2 章「演算と型」, 繰返し等)・・・ 10/23
- 条件分岐 (3 章「プログラムの流れの分岐」)・・・ 10/31(火) は休講 ,11/6
- 繰返し (4 章「プログラムの流れの繰返し」)・・・ 11/13
- 配列, 関数, サブルーチン 1 (5 章「配列」6 章「関数」)・・・ 11/20
- 配列, 関数, サブルーチン 2 (5 章「配列」6 章「関数」)・・・ 11/27
- 基本型 (7 章「基本型」)・・・ 12/4
- 文字列 (9 章「文字列の基本」)・・・ 12/11
- ポインタ 1 (10 章「ポインタ」)・・・ 12/18
- ポインタ 2 (10 章「ポインタ」)・・・ 12/25
- ポインタ 3 (11 章「文字列とポインタ」)・・・ 1/15
- 構造体 (C, Fortran) (12 章「構造体」)・・・ 1/22
- ファイル入出力 (13 章「ファイル入出力」, 自由課題)・・・ 1/29
- まとめ・アンケート, 自由課題・・・ 2/5

戦略ジャンケン 2

前回と異なる戦略 (学習する) のじゃんけんプログラムを作ろう。

配列を使ってプログラムを書こう (配列を使わない例を janken2.c として web にあげました)

```
        if(j > 2) {
            if(human_prev==0){
                if(gg > gc && gg > gp)
                    comp=2;
                else if (gc > gp)
                    comp=0;
            }
            else
                comp=1;
        }
```

```
if(human_prev==1){
    if(cc > cg && cc > cp)
        comp=0;
    else if (cg > cp)
        comp=2;
    else
        comp=1;
}
```

```
if(human_prev==2){
    if(pp > pg && pp > pc)
        comp=1;
    else if (pg > pc)
        comp=2;
    else
        comp=1;
}
```

(中略)

```
if (human_prev==0 && human==0) gg++;
if (human_prev==0 && human==1) gc++;
if (human_prev==0 && human==2) gp++;
if (human_prev==1 && human==0) cg++;
if (human_prev==1 && human==1) cc++;
if (human_prev==1 && human==2) cp++;
if (human_prev==2 && human==0) pg++;
if (human_prev==2 && human==1) pc++;
if (human_prev==2 && human==2) pp++;
    human_prev=human;
```

array1d.f90

```
PROGRAM array
  IMPLICIT NONE
  INTEGER:: I, J
  INTEGER, DIMENSION(10) :: P
  DO I=1,10
    P(I)=I*10
  ENDDO
  DO J=1,10
    WRITE(*,*) P(J)
  ENDDO
END PROGRAM array
```

dotproduct.f90

```
PROGRAM dotproduct
  IMPLICIT NONE
  INTEGER, DIMENSION(3) :: A=(/1,2,3/),B=(/3,2,1/)
  INTEGER:: C
  C=DOT_PRODUCT(A,B)
  WRITE(*,*) C
END PROGRAM dotproduct
```

matmultest.f90

```
PROGRAM matmultest
  IMPLICIT NONE
  INTEGER, DIMENSION(1,3) :: A=RESHAPE((/1,2,3/),(/1,3/))
  INTEGER, DIMENSION(3,1) :: B=RESHAPE((/3,2,1/),(/3,1/))
  INTEGER, DIMENSION(1,1) :: C
  C=MATMUL(A,B)
  WRITE(*,*) C
END PROGRAM matmultest
```

Fortran

型名, DIMENSION(size1,size2, ...) :: 配列名

初期化: 型名, DIMENSION(size) :: 配列名=(/ a₁, a₂, a₃, ... /)

- 1次元配列: REAL, DIMENSION(10) :: P
- 1次元配列 (下限あり): REAL, DIMENSION(0:9) :: P
- 2次元配列: REAL, DIMENSION(10,10) :: Q
- 3次元配列: REAL, DIMENSION(10,10,10) :: R

C 言語

型名, 配列名 [size1][size2][...]]

初期化: 型名 配列名 [size1]={a₁, a₂, a₃, ...}

Fortran との違い

- (1) 使用できる最大の数字=size-1, 配列添え字は 0 からスタート。
 - (2) 動的に size は決められない: malloc 関数を用いる必要がある。
 - (3) 配列の size をオーバーして参照してもエラーがでない。エラー処理必要。
- 1次元配列: double p[10]
 - 2次元配列: double q[10][10]
 - 3次元配列: double r[10][10][10]

data2.c

初期化しなかったらどうなる？

配列の size を超えて参照されたらどうなる？

```
#include <stdio.h>
int main(void)
{
    int i;
    int dat[5];
    for (i=0; i< 6; i++){
        printf("dat[%d] %d %p\n\n",i,dat[i], &dat[i]);
    }
    return 0;
}
```

配列の例 (C 言語)

data2.c

```
#include <stdio.h>
int main(void)
{
    int i;
    int dat[5];
    for (i=0; i< 6; i++){
        printf("dat[%d] %d %p\n\n",i,dat[i], &dat[i]);
    }
    return 0;
}
```

結果

```
dat[0] 0 0x7fff52baf800
dat[1] 0 0x7fff52baf804
dat[2] 0 0x7fff52baf808
dat[3] 0 0x7fff52baf80c
dat[4] 1387984944 0x7fff52baf810
dat[5] 32767 0x7fff52baf814
```

サブルーチン (Fortran)

関数, サブルーチンが書けるようになるとブロック化が可能になり, 再利用性が向上し, 大規模なプログラム作成が容易になる.

サブルーチン

```
SUBROUTINE 名前 (仮引数の列)
IMPLICIT NONE
型名, INTENT(IN) :: 入力引数名
型名, INTENT(OUT) :: 出力引数名
型名, INTENT(INOUT) :: 入出力引数名
ローカル変数の型宣言
(処理内容)
RETURN
END SUBROUTINE 名前
```

サブルーチンを呼び出す

```
INTERFACE
  SUBROUTINE 名前
    IMPLICIT NONE
    型名, INTENT(IN) :: 入力引数名
    ... (サブルーチン内と同じにする)
  END SUBROUTINE 名前
END INTERFACE
と呼び出す主プログラム等の最初に書いておき, CALL 名前 (実引数) とする。
```

サブルーチンの例 (Fortran)

testsr.f90

```
PROGRAM testsr
  IMPLICIT NONE
  INTEGER :: A,B,C
  INTERFACE
    SUBROUTINE SEKI(E,F,G)
      IMPLICIT NONE
      INTEGER, INTENT(IN) :: E,F
      INTEGER, INTENT(OUT) :: G
    END SUBROUTINE SEKI
  END INTERFACE
  READ(*,*) A,B
  CALL SEKI(A,B,C)
  WRITE(*,*) 'C=',C
  STOP
END PROGRAM testsr

SUBROUTINE SEKI(E,F,G)
  IMPLICIT NONE
  INTEGER, INTENT(IN) :: E,F
  INTEGER, INTENT(OUT) :: G
  G=E*F
  RETURN
END SUBROUTINE SEKI
```

関数の例 (Fortran)

testfc.f90

```
PROGRAM testfc
  IMPLICIT NONE
  INTEGER :: A,B,C
  INTERFACE
    FUNCTION SEKI(E,F)
      IMPLICIT NONE
      INTEGER SEKI
      INTEGER, INTENT(IN) :: E,F
    END FUNCTION SEKI
  END INTERFACE
  READ(*,*) A,B
  C=SEKI(A,B)
  WRITE(*,*) 'C=',C
  STOP
END PROGRAM testfc

FUNCTION SEKI(E,F)
  IMPLICIT NONE
  INTEGER :: SEKI
  INTEGER, INTENT(IN) :: E,F
  SEKI=E*F
  RETURN
END FUNCTION SEKI
```

関数 (C 言語)

```
結果の型 関数名 (引数の型 引数名) {  
  ...  
  (処理内容)  
  return 関数値}
```

関数を使う側 (C 言語)

関数の型 関数名 (引数の型) ; が必要。ただし、関数の定義を先に書いて、その後関数を使用する関数、メインプログラムを書くと、関数の型宣言は省略できる (教科書の例題)。

testfc.c

```
#include <stdio.h>
int main(void)
{
    double x,y;
    double f(double);
    printf("x=\n");
    scanf("%lf",&x);
    y=f(x);
    printf("f(%lf)=%lf\n",x,y);
}

double f(double x){
    double z;
    z=x*x*x;
    return z;
}
```

6 回目レポートと次回の小テスト

問題 1-5 のうち 3 つを選んでレポートとして (問題 4,5 で明解 C(5 章,6 章の全演習問題をやっても可) アカサスポータルから提出してください。

うまく提出できない場合はメール添付で

ishii@cphys.s.kanazawa-u.ac.jp へ提出してください。

次回の小テストは明解 C 言語 5 章と講義ノート 6 回目の内容から。

問題 1

戦略じゃんけんの異なる戦略によるプログラム作成。

問題 2

C 言語でベクトルを配列で関数に渡して, 外積, 内積を計算する関数を使うプログラム。

問題 3

n 次の正則行列の直積 $\mathbf{C} = \mathbf{A} \otimes \mathbf{B}$ を計算するプログラム。 \mathbf{C} の要素は $C_{\alpha,\beta} = A_{ij}B_{kl}$ の様に計算され, $\alpha = n(i-1) + k$, $\beta = n(j-1) + l$, \mathbf{C} は $2n \times 2n$ の正則行列になる。

6 回目レポート (つづき)

問題 4: プログラムを C ならびに Fortran90 で作成せよ。

明解 C(5 章) の演習問題の奇数番。

問題 5: プログラムを C ならびに Fortran90 で作成せよ。

明解 C(6 章) の演習問題の奇数番。