プログラミング基礎

2023年度1Q　火曜日３,４時限(10:45~12:25)
金曜日1~４時限(　8:50~12:25)

工学院　情報通信系

中山実，渡辺義浩
伊藤泉，小杉哲
TA:小泊大輝，千脇彰悟

4/28(金) 8:50~12:25

- 第6回「配列 2」

1. 続・配列の使い方
2. 2次元配列

配列の利用（度数分布の作成）

- 数字を入力
- 0より小さい値、100より大きい値が入力された時点で入力終了

- 11のRankに分類し，それぞれの度数を出力する.
  - Rank 0: 0～9点,
  - Rank 1: 10~19点,
  - …
  - Rank 9: 90~99点,
  - Rank 10: 100点

実行結果例

```
15
45
88
…
110
Rank 0, Freq 0
Rank 1, Freq 2
Rank 2, Freq 0
Rank 3, Freq 8
Rank 4, Freq 10
Rank 5, Freq 12
Rank 6, Freq 5
Rank 7, Freq 1
Rank 8, Freq 4
Rank 9, Freq 0
Rank 10, Freq 0
```

配列の利用（度数分布の作成）

- マクロを使って定数を定義
- 配列のサイズは下記でも指定可能（仕様による）

```
int num;
scanf("%d",&num);

int table[num];
```

```
#include <stdio.h>
#define MAX 100
#define MIN 0
#define RANK 11
#define STEP 10

int main (void) {
    double score;
    int i, table[RANK];
    for (i=0; i < RANK; i++) table[i] = 0;

    while (1) {
        scanf ("%lf", &score);
        if (score < MIN || score > MAX){
            break;
        }
        i = (int)(score / STEP);
        table[i] ++;
    }
    for (i=0; i < RANK; i++){
        printf ("Rank %d, Freq %d¥n", i, table[i]);
    }
    return 0;
}
```

配列の利用（度数分布の作成）

- 配列内の数を 0 に初期化
- 下記でも可

```
int table[RANK]={};
```

- double 変数を読み込む場合、変換仕様は%lfとする

```
#include <stdio.h>
#define MAX 100
#define MIN 0
#define RANK 11
#define STEP 10

int main (void) {
    double score;
    int i, table[RANK];
    for (i=0; i < RANK; i++) table[i] = 0;

    while (1) {
        scanf ("%lf", &score);
        if (score < MIN || score > MAX){
            break;
        }
        i = (int)(score / STEP);
        table[i] ++;
    }
    for (i=0; i < RANK; i++){
        printf ("Rank %d, Freq %d¥n", i, table[i]);
    }
    return 0;
}
```

配列の利用（度数分布の作成）

- 意図的な無限ループ
- 0未満，101以上のscore入力でループから抜ける

```c
#include <stdio.h>
#define MAX 100
#define MIN 0
#define RANK 11
#define STEP 10

int main (void) {
    double score;
    int i, table[RANK];
    for (i=0; i < RANK; i++) table[i] = 0;

    while (1) {
        scanf ("%lf", &score);
        if (score < MIN || score > MAX){
            break;
        }
        i = (int)(score / STEP);
        table[i] ++;
    }
    for (i=0; i < RANK; i++){
        printf ("Rank %d, Freq %d¥n", i, table[i]);
    }
    return 0;
}
```

配列の利用（度数分布の作成）

- 点をSTEPで割った値でRankを決める.

- (int)は型キャスト
- 変数の種類を変換する
- 実数を整数に変換し，小数点以下を削除する

```
#include <stdio.h>
#define MAX 100
#define MIN 0
#define RANK 11
#define STEP 10

int main (void) {
    double score;
    int i, table[RANK];
    for (i=0; i < RANK; i++) table[i] = 0;

    while (1) {
        scanf ("%lf", &score);
        if (score < MIN || score > MAX){
            break;
        }
        i = (int)(score / STEP);
        table[i] ++;
    }
    for (i=0; i < RANK; i++){
        printf ("Rank %d, Freq %d¥n", i, table[i]);
    }
    return 0;
}
```

配列の利用（度数分布の作成）

- 度数分布の出力

```c
#include <stdio.h>
#define MAX 100
#define MIN 0
#define RANK 11
#define STEP 10

int main (void) {
    double score;
    int i, table[RANK];
    for (i=0; i < RANK; i++) table[i] = 0;

    while (1) {
        scanf ("%lf", &score);
        if (score < MIN || score > MAX){
            break;
        }
        i = (int)(score / STEP);
        table[i] ++;
    }
    for (i=0; i < RANK; i++){
        printf ("Rank %d, Freq %d¥n", i, table[i]);
    }
    return 0;
}
```

配列の利用（度数分布の作成）

- 110が入力された時点で入力終了

- 11のRankに分類し，それぞれの度数を出力する.
  - Rank 0: 0～9点,
  - Rank 1: 10~19点,
  - …
  - Rank 9: 90~99点,
  - Rank 10: 100点

実行結果例

```
15
45
88
…
110
Rank 0, Freq 0
Rank 1, Freq 2
Rank 2, Freq 0
Rank 3, Freq 8
Rank 4, Freq 10
Rank 5, Freq 12
Rank 6, Freq 5
Rank 7, Freq 1
Rank 8, Freq 4
Rank 9, Freq 0
Rank 10, Freq 0
```

配列の利用（条件に合うデータの抽出）

- 入力された月の日数を出力
- 31日の月の1年の総日数を出力

実行結果例

```
Month?: 3
The number of days in Month 3 is 31
Total days of months with 31days: 217
```

配列の利用（条件に合うデータの抽出）

- 配列の宣言と初期化

```
#include <stdio.h>

int main(void) {
    int i, month, totaldays = 0;
    int daysinmonth[12]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    printf("Month?: ");
    scanf("%d", &month);
    printf("The number of days in Month %d is %d¥n", month,
daysinmonth[month-1]);
    for(i=0; i<12; i++){
        if ( daysinmonth[i] == 31)
            totaldays +=31;
    }
    printf("Total days of months with 31days: %d", totaldays);
    return 0;
}
```

配列の利用（条件に合うデータの抽出）

- 指定された月の日にちを出力

```
#include <stdio.h>

int main(void) {
    int i, month, totaldays = 0;
    int daysinmonth[12]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    printf("Month?: ");
    scanf("%d", &month);
    printf("The number of days in Month %d is %d¥n", month,
daysinmonth[month-1]);
    for(i=0; i<12; i++){
        if ( daysinmonth[i] == 31)
            totaldays +=31;
    }
    printf("Total days of months with 31days: %d", totaldays);
    return 0;
}
```

配列の利用（条件に合うデータの抽出）

- 31日の月かどうかの判別
- 31日の月であれば日数を加算

```
#include <stdio.h>

int main(void) {
    int i, month, totaldays = 0;
    int daysinmonth[12]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    printf("Month?: ");
    scanf("%d", &month);
    printf("The number of days in Month %d is %d¥n", month,
daysinmonth[month-1]);
    for(i=0; i<12; i++){
        if ( daysinmonth[i] == 31)
            totaldays +=31;
    }
    printf("Total days of months with 31days: %d", totaldays);
    return 0;
}
```

配列の利用（条件に合うデータの抽出）

- totaldays += 31*(daysinmonth[i] == 31);でも可
- 論理演算の結果が真の時，「１」で表現される．
- 論理演算の結果が偽の時，「０」で表現される．

```
#include <stdio.h>

int main(void) {
    int i, month, totaldays = 0;
    int daysinmonth[12]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    printf("Month?: ");
    scanf("%d", &month);
    printf("The number of days in Month %d is %d¥n", month,
daysinmonth[month-1]);
    for(i=0; i<12; i++){
        if ( daysinmonth[i] == 31)
            totaldays +=31;
    }
    printf("Total days of months with 31days: %d", totaldays);
    return 0;
}
```

配列の利用（条件に合うデータの抽出）

- 31日の月の総日数を出力

```
#include <stdio.h>

int main(void) {
    int i, month, totaldays = 0;
    int daysinmonth[12]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    printf("Month?: ");
    scanf("%d", &month);
    printf("The number of days in Month %d is %d¥n", month,
daysinmonth[month-1]);
    for(i=0; i<12; i++){
        if ( daysinmonth[i] == 31)
            totaldays +=31;
    }
    printf("Total days of months with 31days: %d", totaldays);
    return 0;
}
```

配列の利用（条件に合うデータの抽出）

実行結果例

```
Month?: 3
The number of days in Month 3 is 31
Total days of months with 31days: 217
```

```c
#include <stdio.h>

int main(void) {
    int i, month, totaldays = 0;
    int daysinmonth[12]={31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
    printf("Month?: ");
    scanf("%d", &month);
    printf("The number of days in Month %d is %d¥n", month,
daysinmonth[month-1]);
    for(i=0; i<12; i++){
        if ( daysinmonth[i] == 31)
            totaldays +=31;
    }
    printf("Total days of months with 31days: %d", totaldays);
    return 0;
}
```

配列の利用（入力，コピー）

- 10名の学生の点数を入力
- 最高点と最低点を出力
- 0番目と9番目の点数を表示

実行結果例

```
Student[0]: Score? 10
Student[1]: Score? 20
Student[2]: Score? 30
Student[3]: Score? 40
Student[4]: Score? 50
Student[5]: Score? 60
Student[6]: Score? 70
Student[7]: Score? 80
Student[8]: Score? 90
Student[9]: Score? 100
max: 100, min: 10
score2[0]: 10, score2[9]: 100
```

- マクロを使って，定数を定義

```c
#include <stdio.h>
#define Number 10

int main(void) {
    int i;
    int max, min;
    int score1[Number], score2[Number];

    for(i=0; i < Number; i++){
        printf("Student[%d]: Score? ¥n",i);
        scanf("%d", &score1[i]);
    }
    max=min=score1[0];
    for(i=1; i < Number; i++){
        if(score1[i] > max) max=score1[i];
        if(score1[i] < min) min=score1[i];
    }
    printf("max: %d, min: %d¥n", max, min);

    for(i=0; i < Number; i++) score2[i]=score1[i];
    printf("score2[0]: %d, score2[9]: %d¥n", score2[0],score2[9]);
    return 0;
}
```

- 得点データを入力する配列の宣言

```
#include <stdio.h>
#define Number 10

int main(void) {
    int i;
    int max, min;
    int score1[Number], score2[Number];

    for(i=0; i < Number; i++){
        printf("Student[%d]: Score? ¥n",i);
        scanf("%d", &score1[i]);
    }
    max=min=score1[0];
    for(i=1; i < Number; i++){
        if(score1[i] > max) max=score1[i];
        if(score1[i] < min) min=score1[i];
    }
    printf("max: %d, min: %d¥n", max, min);

    for(i=0; i < Number; i++) score2[i]=score1[i];
    printf("score2[0]: %d, score2[9]: %d¥n", score2[0],score2[9]);
    return 0;
}
```

- 配列の各要素にデータを読み込む

```
#include <stdio.h>
#define Number 10

int main(void) {
    int i;
    int max, min;
    int score1[Number], score2[Number];

    for(i=0; i < Number; i++){
        printf("Student[%d]: Score? ¥n",i);
        scanf("%d", &score1[i]);
    }
    max=min=score1[0];
    for(i=1; i < Number; i++){
        if(score1[i] > max) max=score1[i];
        if(score1[i] < min) min=score1[i];
    }
    printf("max: %d, min: %d¥n", max, min);

    for(i=0; i < Number; i++) score2[i]=score1[i];
    printf("score2[0]: %d, score2[9]: %d¥n", score2[0],score2[9]);
    return 0;
}
```

- 最大，最小の検索

```c
#include <stdio.h>
#define Number 10

int main(void) {
    int i;
    int max, min;
    int score1[Number], score2[Number];

    for(i=0; i < Number; i++){
        printf("Student[%d]: Score? ¥n",i);
        scanf("%d", &score1[i]);
    }
    max=min=score1[0];
    for(i=1; i < Number; i++){
        if(score1[i] > max) max=score1[i];
        if(score1[i] < min) min=score1[i];
    }
    printf("max: %d, min: %d¥n", max, min);

    for(i=0; i < Number; i++) score2[i]=score1[i];
    printf("score2[0]: %d, score2[9]: %d¥n", score2[0],score2[9]);
    return 0;
}
```

- データのコピー

```c
#include <stdio.h>
#define Number 10

int main(void) {
    int i;
    int max, min;
    int score1[Number], score2[Number];

    for(i=0; i < Number; i++){
        printf("Student[%d]: Score? ¥n",i);
        scanf("%d", &score1[i]);
    }
    max=min=score1[0];
    for(i=1; i < Number; i++){
        if(score1[i] > max) max=score1[i];
        if(score1[i] < min) min=score1[i];
    }
    printf("max: %d, min: %d¥n", max, min);

    for(i=0; i < Number; i++) score2[i]=score1[i];
    printf("score2[0]: %d, score2[9]: %d¥n", score2[0],score2[9]);
    return 0;
}
```

× score2=score1;
配列の代入は不可

```c
#include <stdio.h>
#define Number 10

int main(void) {
    int i;
    int max, min;
    int score1[Number], score2[Number];

    for(i=0; i < Number; i++){
        printf("Student[%d]: Score? ¥n",i);
        scanf("%d", &score1[i]);
    }
    max=min=score1[0];
    for(i=1; i < Number; i++){
        if(score1[i] > max) max=score1[i];
        if(score1[i] < min) min=score1[i];
    }
    printf("max: %d, min: %d¥n", max, min);

    for(i=0; i < Number; i++) score2[i]=score1[i];
    printf("score2[0]: %d, score2[9]: %d¥n", score2[0],score2[9]);
    return 0;
}
```

実行結果例

```
Student[0]: Score? 10
Student[1]: Score? 20
Student[2]: Score? 30
Student[3]: Score? 40
Student[4]: Score? 50
Student[5]: Score? 60
Student[6]: Score? 70
Student[7]: Score? 80
Student[8]: Score? 90
Student[9]: Score? 100
max: 100, min: 10
score2[0]: 10, score2[9]: 100
```

4/28(金) 8:50~12:25

- 第6回「配列 2」

1. 続・配列の使い方
2. 2次元配列

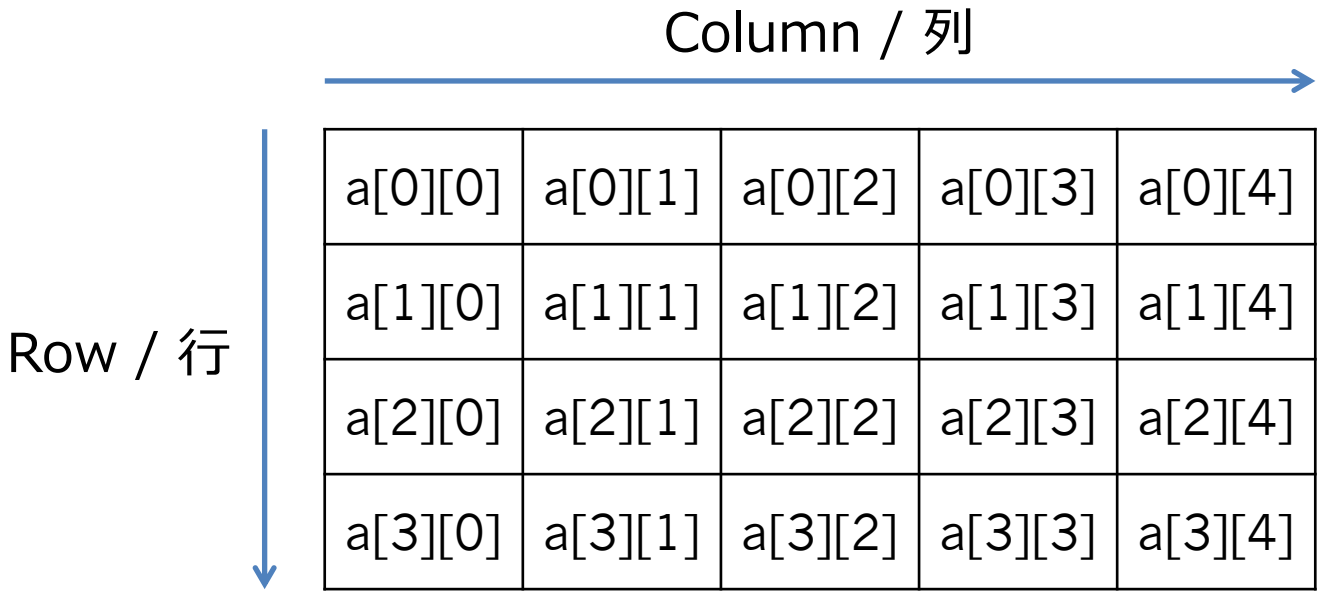
二次元配列

- `int a[4][5];`

a[0][0]	a[0][1]	a[0][2]	a[0][3]	a[0][4]
a[1][0]	a[1][1]	a[1][2]	a[1][3]	a[1][4]
a[2][0]	a[2][1]	a[2][2]	a[2][3]	a[2][4]
a[3][0]	a[3][1]	a[3][2]	a[3][3]	a[3][4]

- メモリ上に4×5=20個分の領域が確保される
- 実際には計算機のメモリ上には一次元的に連続した領域に並んでいる
  - 連続したアドレスが振られている
  - `a[0][0]→a[0][1] →…a[1][0]→a[1][1]→…→a[3][4]`
  - 末尾側の添え字が優先的に増えていく順にならぶ

二次元配列


Column / 列
Row / 行
a[0][0] a[0][1] a[0][2] a[0][3] a[0][4]
a[1][0] a[1][1] a[1][2] a[1][3] a[1][4]
a[2][0] a[2][1] a[2][2] a[2][3] a[2][4]
a[3][0] a[3][1] a[3][2] a[3][3] a[3][4]

- 二次元配列の宣言と初期化

```c
#include <stdio.h>
#define ROW 3
#define COLUMN 2

int main(void) {
    int i, j;
    int matrix[ROW][COLUMN]={ {0,1}, {2,3}, {2,4} };

    for(i=0; i < ROW; i++){
        for(j=0; j < COLUMN; j++){
            printf("matrix[%d][%d]= %d¥n", i, j, matrix[i][j]);
        }
    }
    return 0;
}
```

- 二次元配列の各要素を順番に表示

```c
#include <stdio.h>
#define ROW 3
#define COLUMN 2

int main(void) {
    int i, j;
    int matrix[ROW][COLUMN]={ {0,1}, {2,3}, {2,4} };

    for(i=0; i < ROW; i++){
        for(j=0; j < COLUMN; j++){
            printf("matrix[%d][%d]= %d¥n", i, j, matrix[i][j]);
        }
    }
    return 0;
}
```

実行結果

```
matrix[0][0]= 0
matrix[0][1]= 1
matrix[1][0]= 2
matrix[1][1]= 3
matrix[2][0]= 2
matrix[2][1]= 4
```

二次元配列の利用

- 学生の科目ごとの得点を集めて、0番目の科目の最高点と最低点を出力

実行結果例

```
Student[0], Subject[0]: Score? 10
Student[0], Subject[1]: Score? 20
Student[0], Subject[2]: Score? 30
Student[1], Subject[0]: Score? 40
Student[1], Subject[1]: Score? 50
Student[1], Subject[2]: Score? 60
Student[2], Subject[0]: Score? 70
Student[2], Subject[1]: Score? 80
Student[2], Subject[2]: Score? 90
Student[3], Subject[0]: Score? 100
Student[3], Subject[1]: Score? 110
Student[3], Subject[2]: Score? 120
max_Subject[0]: 100, min_Subject[0]: 10
```

```c
#include <stdio.h>
#define Student 4
#define Subject 3

int main(void) {
    int i,j;
    int max, min;
    int score[Student][Subject];

    for(i=0; i < Student; i++){
        for(j=0; j < Subject; j++){
            printf("Student[%d], Subject[%d]: Score? ",i,j);
            scanf("%d", &score[i][j]);
        }
    }
    max=min=score[0][0];
    for(i=0;i < Student;i++){
        if(score[i][0] > max) max=score[i][0];
        if(score[i][0] < min) min=score[i][0];
    }
    printf("max_Subject[0]: %d, min_Subject[0]: %d¥n", max, min);

    return 0;
}
```

学生数，科目数

得点を格納する二次元配列

```c
#include <stdio.h>
#define Student 4
#define Subject 3

int main(void) {
    int i,j;
    int max, min;
    int score[Student][Subject];

    for(i=0; i < Student; i++){
        for(j=0; j < Subject; j++){
            printf("Student[%d], Subject[%d]: Score? ",i,j);
            scanf("%d", &score[i][j]);
        }
    }
    max=min=score[0][0];
    for(i=0;i < Student;i++){
        if(score[i][0] > max) max=score[i][0];
        if(score[i][0] < min) min=score[i][0];
    }
    printf("max_Subject[0]: %d, min_Subject[0]: %d¥n", max, min);

    return 0;
}
```

配列の各要素に得点を格納

```c
#include <stdio.h>
#define Student 4
#define Subject 3

int main(void) {
    int i,j;
    int max, min;
    int score[Student][Subject];

    for(i=0; i < Student; i++){
        for(j=0; j < Subject; j++){
            printf("Student[%d], Subject[%d]: Score? ",i,j);
            scanf("%d", &score[i][j]);
        }
    }
    max=min=score[0][0];
    for(i=0;i < Student;i++){
        if(score[i][0] > max) max=score[i][0];
        if(score[i][0] < min) min=score[i][0];
    }
    printf("max_Subject[0]: %d, min_Subject[0]: %d¥n", max, min);

    return 0;
}
```

Subject[0]の中で最大，最小の検索

二次元配列の利用

- 二次元配列に文字列を格納して、1文字目と2文字目を表示

実行結果

```
The word is Giants
The 1st and 2nd letters are: G i
The word is Swallows
The 1st and 2nd letters are: S w
The word is Tigers
The 1st and 2nd letters are: T i
```

- 文字型二次元配列の宣言，初期化
- 宣言時に要素数を省略することも可能
- char str3[][20]={“Giants”, “Swallows”, “Tigers”};

```
#include <stdio.h>

int main(void) {
    int i;
    char str3[3][20]={"Giants", "Swallows", "Tigers"};

    for (i=0; i<3; i++){
        printf("The word is %s ¥n",str3[i]);
        printf("The 1st and 2nd letters are: %c %c ¥n",str3[i][0],str3[i][1]);
    }
    return 0;
}
```

- 1文字の出力：%c
- 文字列全体の出力：%s

```c
#include <stdio.h>

int main(void) {
    int i;
    char str3[3][20]={"Giants", "Swallows", "Tigers"};

    for (i=0; i<3; i++){
        printf("The word is %s ¥n",str3[i]);
        printf("The 1st and 2nd letters are: %c %c ¥n",str3[i][0],str3[i][1]);
    }
    return 0;
}
```

```c
#include <stdio.h>

int main(void) {
    int i;
    char str3[3][20]={"Giants", "Swallows", "Tigers"};

    for (i=0; i<3; i++){
        printf("The word is %s ¥n",str3[i]);
        printf("The 1st and 2nd letters are: %c %c ¥n",str3[i][0],str3[i][1]);
    }
    return 0;
}
```

実行結果

```
The word is Giants
The 1st and 2nd letters are: G i
The word is Swallows
The 1st and 2nd letters are: S w
The word is Tigers
The 1st and 2nd letters are: T i
```