Introduction to Programming

— Arrays·Macro(#define) —

Waseda University

Today's topics

- How to use arrays properly.
  - Declaring of arrays
  - Subscript operator "[ ]"
  - Scope of the index
  - Initialization of arrays
  - Multidimensional array
- Usage of a constant value by using `#define`
  - How to define and use

Arrays

Arrays (declaring an array of 100 int type variables)

```
int Data[100];          /*Declaring arrays*/
```

- We use the array when you have to repeat variables handled in the same way.
- Each element of an array is distinguished by the index (the number inside the square brackets[ ])
- We reference an element of an array by using subscript operator [ ].
- Note that the index number starts from 0 and the last index number in an array of 100 is $100 - 1 = 99$.
- An assignment is the same as we've learned before.

```
Data[0]=3;
Data[10]=2;
```

Initialization of the array

- We can initialize at the same time as declaring an array.

```
int Data[5]={23,34,45,68,41};
```

- We can only collectively initialize when declaring.

```
int Data[5];
Data = {23,34,45,68,41};
```

- if you're going to initialize after declaring:

```
int Data[5];
Data[0] = 23 ;
Data[1] = 34 ;
...
```

Usage of the array

This program uses an array with 3 int type components (data3.c)

```
#include <stdio.h>

int main(void){
  int Data[3];                          /* Declaring an array */
  Data[0]=10;                           /* Assignment*/
  Data[1]=Data[0]*2;                    /*Calculation, assignment*/
  Data[2]=Data[1]+3;                    /*Calculation, assignment*/
  printf("Data=%d, %d.¥n",Data[0],Data[2]);
  return 0;
}
```

- Pay attention to the fact that here the declared array subscript is "from 0 to 2".

Example 1: This program by using an array:

To input the test scores for 3 subjects and calculating the sum (array.c)

```
#include <stdio.h>
int main(void){
  int Scores[3],Sum=0, i;                  /*Declaring an array*/
  for(i=0;i<3;i++){                        /* for loop */
    printf("Input score %d:",i);           /* Display */
    scanf("%d",&Scores[i]);                /* Input  */
  }
  for(i=0;i<3;i++){                        /* for loop */
    Sum += Scores[i];                      /* Add */
  }
  printf("Total is %d.¥n",Sum);            /* Display */
  return 0;
}
```

- Copy this program and execute it.
- We call this program "array.c".

Usage of the array

- Even if the number of variables increases, all you have to do is to change the index number of an array.
- If you don't use an array, you have to do the following:

```
 int Score0,Score1,Score2,Score3,…,Sum=0, i;
 printf("Input score 0:");
 scanf("%d", &Scores0);
 printf("Input score 1:");
 scanf("%d", &Scores1);
 printf("Input score 2:");
 scanf("%d", &Scores2);
 printf("Input score 3:");
 scanf("%d", &Scores3);
…
```

Macro definitions (#define)

- With example array.c using the array, the value "3" has appeared many times.
- It takes time to write "3" many times and makes mistakes.
- It is easy to modify a program after you define some string as 3 by using "#define"
- You can check typos when compiling.

Macro definitions (#define string1 string2)

We can substitute string1 with string2.

```
#define SIZE 3
```

- After this line, "SIZE" is regarded as "3".
- We recommend to write comments before macro definitions.

- It is easy to read, because of giving a macro string name for a constant value.
- Usually, a macro string name is written in capital letters.

Example 2: The program by using `#define`

Change "3" subjects to "SIZE" in example 1. (array2.c)

```
#include <stdio.h>
#define SIZE 3                                /*Macro definition*/
int main(void){
  int Scores[SIZE],Sum=0, i;                  /*Declaring an array*/
  for(i=0; i<SIZE;i++){                       /* for loop */
    printf("Input score %d:",i);              /*  Display */
    scanf("%d", &Scores[i]);                  /*  Input  */
  }
  for(i=0; i<SIZE;i++)                        /* for loop */
    Sum += Scores[i];                         /*  Add */
  printf("Total is %d,¥n", Sum);              /*  Display  */
  return 0;
}
```

- First, Copy array.c to array2.c
- Rewrite "3" subjects to "SIZE" by using macro definition and execute it.

Example 3

Example 3: score.c

- File name is "score.c".
- Assign scores for 20 peoples to an array,
- Scores are calculated by the following:

  ith people scores = `(i*83+11)%101` **(**$i = 0, 1, ..., 19$**)**

- Use macro definition for "20".
- Next, display all scores separated by comma.
- Finally, display a maximum score.
- For example:

```
12, 95, 77,..., 0,
Highest score is ???.
```

Hint of example 3

- How to choose a maximum value
  - Assign first value of the array to the variable **max**
  - Compare the given values and set the greater value in order to get the maximum value.
  - The remaining part writes it as follows:

```
max =Scores[0];
for(i=1;…;…){
  if(max <…) …
  printf("Highest score is %3d.¥n ",…);
}
```

- Don't forget to use macro definition `#define` in order to replace "20" with "SIZE"

Multidimensional array

- Array can have more than one dimension.
- For example, the declaration for a two-dimensional array is:

```
int Scores[50][5];
for (i=0; i<50; i++){
  printf("Student %d:¥n",i);
  for(j=0;j<5;j++){
    printf("Input subject %d:",j);
    scanf("%d",&Scores[i][j]);
  }
}
```

- This is an example of the results for a 50-person, 5-subject score using a 2-dimensional array.
- When we use a large array, we have to use other methods.

Summary

- How to use arrays properly.
  - Declaring of arrays
  - Subscript operator "[ ]"
  - Scope of the index
  - Initialization of arrays
  - Multidimensional array
- Usage of a constant value by using `#define preprocessor`
  - How to define and use