# C PROGRAMMING INTRODUCTION

WEEK 10: ARRAYS

Arrays

- A block of many variables of the same type
- Array can be declared for any type
  - E.g. `int Arr[10]` is an array of 10 integers.
- Examples:
  - list of students' marks
  - series of numbers entered by user
  - vectors
  - matrices

Arrays in Memory

- Sequence of variables of specified type
- The array variable itself holds the address in memory of beginning of sequence
- Example:

`int arr[10];`

0	1	2	3	4	5	6	7	8	9

arr
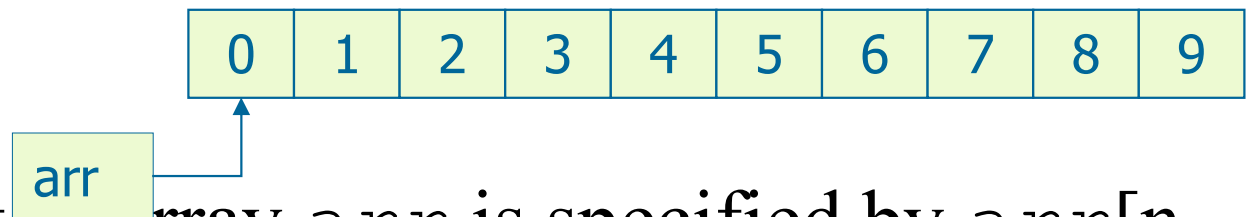
- The n-th element of array `arr` is specified by `arr[n-1]` **(0-based)**

Initialization

- Like in the case of regular variables, we can initialize the array during declaration.

- the number of initializers cannot be more than the number of elements in the array

  - but it can be less

  - in which case, the remaining elements are initialized to 0

Initialization

- if you like, the array size can be inferred from the number of initializers **by leaving the square brackets empty**
  - so these are identical declarations :
    - int array1 [8] = {2, 4, 6, 8, 10, 12, 14, 16};
    - int array2 [] = {2, 4, 6, 8, 10, 12, 14, 16};

Example: Initialization with for statement

- Create an array of even numbers from 2 to 20. Print the content of this array.

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

int main()
{
        int s[ arraySize ];  // array S has 10 elements
        int i;
       for ( i = 0; i < arraySize; i++ )
            s[ i ] = 2 + 2 * i;
       printf("Element \t Value\n");
       for ( i = 0; i < arraySize; i++ )
                printf("%d\t%d\n", i, s[i]);
       return 0;
}
```

Data input and output for array

- Running a for loop, in each loop:
  - use data input functions like scanf
- or
  - use data output functions like printf

month	rainfall (in mm)
1	40
2	45
3	95
4	130
5	220
6	210
7	185
8	135
9	80
10	40
11	45
12	30

table of rainfall

Example

```c
#include <stdio.h>
#define  MONTHS 12

/* store and display rainfall in all months of the year */

int main()
{
    int rainfall[MONTHS];
   int i;

   for ( i=0; i < MONTHS; i++ ){
       printf("Enter the rainfall(mm):"); scanf("%d", &rainfall[i] );
   }
   /* Print from January to December */
   for ( i=0; i < MONTHS; i++ )  {
       printf( "%5d " , rainfall[i]);
   }
   printf("\n");
   return 0;
}
```

Exercise 10.1

1) Write a program to input an array that stores 100 integers.

    a) Find the sum of the odd number in the array

    b) Find the minimum value.

Solution 10.1

```c
#include <stdio.h>
#define MAX 100

int main()
{
    int a[MAX];
    int i,s,min;
    for ( i = 0; i < MAX; i++ ){
      printf("a[%d]=",i);scanf("%d", &a[i]);
    }
    s=0; min=a[0];
    for ( i = 0; i < MAX; i++ ){
        if (a[i] < min) min = a[i];
        if (a[i]%2==1) s= s+ a[i];
    }
      printf("\nMinimum value in the array: %d",min);
    printf("Sum of all odd numbers: %d",s);
    return 0;
}
```

Exercise 10.2

- Given an array of which elements are the numbers inputed by user. Find the sum of the local maximum numbers in this array (local maximum element is the element that greater than its two neighbours)

Solution

```
.. // Array data input
sum=0;
for (i=1; i<=size-1; i++)
    if (a[i]>=a[i-1] && a[i]>=a[i+1])
        sum +=a[i];
```

Arrays as function arguments

- Functions can accept arrays as arguments
- Usually the array's size also needs to be passed (why?)

Arrays as function arguments

- For example:
  ```
  int calc_sum(int arr[], int size);
  ```
- Within the function, **arr** is accessed in the usual way
- Changes in the function **affect** the original array!!

Example

```c
int calc_sum(const int arr[], int size)
{
  int i = 0;
  int sum = 0;

  for (i = 0; i < size; ++i)
      sum += arr[i];

  return sum;
}
```

Exercise 10.3

- Implement a function that accepts two integer arrays and returns 1 if they are equal, 0 otherwise
- Write a program that accepts two arrays of integers from the user and checks for equality

Solution

```c
int compare_arrays(const int arr1[], const int
  arr2[], int size)
{
  int i = 0;

  for (i = 0; i < size; ++i)
  {
      if (arr1[i] != arr2[i])
          return 0;
  }


    /* if we got here, both arrays are identical */
    return 1;
}
```

Exercise 10.4

- Write two functions:
  - the first sorts the integers element in an array by the decreasing order.
  - the second sort the odd elements in the decreasing order.
- Write a program that asks user to enter 10 intergers and displays the results after two styles of sorting above.

Solution: Sorting in descending order

```c
void DesSort (int a[], int n)
{
  int tmp;
   for (i = 0; i < n-1 ; i++)
     for (j = i+1; j < n; j++)
       if (a[i] < a [j]) {
                 tmp=a[i];
                 a[i]=a[j];
                 a[j]= tmp;
          }
     }
}
```

Sorting odd numbers

```c
void OddSort (int a[], int n)
{
  int tmp;
   for (i = 0; i < n-1 ; i++)
     for (j = i+1; j < n; j++)
       if (a[i]<a[j] && (a[i]%2) && (a[j]%2))
         {
                    tmp=a[i];
                    a[i]=a[j];
                    a[j]= tmp;

         }

    }
}
```

Exercise 10.5

- Given an integer array:
  a) Count the number of the number 0 in this array.
  b) Find the length of the subsequence that consists the consecutive numbers (all of elements are number 0).
  c) Find the time of appearance of numbers.

Solution for question a)

Traverse the array, when program meet an element 0, increment the counter.

```
count=0;
for (i=1; i<n; i++)
    if (a[i] ==0)
        count++;
```

Solution for question b)

```
int max=0,temp=0;
  for(i=0;i<n;i++)
  {
    if(a[i]==0)
    temp=temp+1;
          if(a[i]!=0){
                if(temp>max){
                    max=temp;
                    temp=0;
                }
          }
  }
  if (temp>max) max=temp;
  printf("Do dai day con bang 0 lon nhat
  la:", max);
```

Solution for question c)

Firstly we sort the array, then count the number of equal elements just after each element.

```c
#include<stdio.h>

void swap(int *a,int *b)
{
  int temp;
  temp=*a;
  *a=*b;
  *b=temp;
}
```

Solution for question c)

```c
void main()
{
  int n,a[100],i,j,dem;
  printf("Enter the number of elements in array:");
  scanf("%d",&n);
  for(i=0;i<n;i++) {
      printf("a[%d] = ",i); scanf("%d",&a[i]);
  }
  for(i=0;i<n-1;i++)
      for(j=i+1;j<n;j++)
              if (a[i]>a[j]) swap(&a[i],&a[j]);

  i=0; j=0; dem=0;
  while (i<n)   {
      dem=0;
      j=i;
      while (a[i]==a[j]) {
              dem++; j++;
      }
      printf("\n%d occurs for %d times in array",a[i],dem);
      i=j;
  }
}
```

Multi-dimensional arrays

- Array of arrays:

```
int A[2][3] = { {1, 2, 3},
                {4, 5, 6} };
```

1	2	3
4	5	6

- Means an array of 2 integer arrays, each of length 3.
- Access: j-th element of the i-array is

```
A[i][j]
```

Example: Matrix addition

```c
#include <stdio.h>

#define SIZE 3

int main()
{
    int A[][SIZE] = {{1,2,3}, {4,5,6}, {7,8,9}};
    int B[][SIZE] = {{1,1,1}, {2,2,2}, {3,3,3}};
    int C[SIZE][SIZE];
    int i = 0, j = 0;

            for (i = 0; i < SIZE; ++i)
               for (j = 0; j < SIZE; ++j)
                  C[i][j] = A[i][j] + B[i][j];

    return 0;
}
```

Exercise 10.6

- Write a program that defines 3 matrices A,B,C of size 3x3 with int elements; initialize the first two matrices (A and B)

- Compute the matrix multiplication of A and B and store it in C (i.e. C = A*B)

- Print all the matrices on the screen

Solution

```c
#include <stdio.h>
void main() {
  float a[3][3], b[3][3], c[3][3];
  int m,n,p;
  int i,j,k;
  float temp;
  for(i=0; i<3; i++){
     for (j=0; j<3; j++) {
        printf("a[%d][%d]=", i+1, j+1);
        scanf("%f", &temp);
        printf("b[%d][%d]=", i+1, j+1);
        scanf("%f", &temp);
        b[i][j]= temp;
     }
   }
```

Solution

```
for(i=0; i<3; i++){
  for (j=0; j<3; j++) {
    c[i][j]=0;
    for (k=0; k<n; k++)
      c[i][j]= c[i][j]+ a[i][k]*b[k][j];
  }
}
printf("\n Matrix A:");
for(i=0; i<3; i++){
  printf("\n");
  for (j=0; j<3; j++) {
    printf("%2.2f\t", a[i][j]);

  }
}
```

Solution

```c
printf("\n Matrix b:");
for(i=0; i<3; i++){
  printf("\n");
  for (j=0; j<3; j++) {
    printf("%2.2f\t", b[i][j]);

  }
}
printf("\n Matrix c:");
for(i=0; i<3; i++){
  printf("\n");
  for (j=0; j<3; j++) {
    printf("%2.2f\t", c[i][j]);
  }
}
```

Exercise 10.7

- Input an array with the number of element n asked from user. Check to see whether the array is symmetric.

Solution

```c
#include <stdio.h>

int checkSymmetric(int a[], int n);

void main()
{
    int a[100],n,i;
    printf(" Number of elements: ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
            {
                        printf("a[%d]= ",i);
                        scanf("%d",&a[i]);
            }
    printf("\n Array's content:\n");
    for(i=0;i<n;i++)
            printf("%d ",a[i]);
    if(checkSymmetric(a,n))
            printf("\n array is symmetric");
    else
            printf("\n array is not symmetric ");
}
```

```c
int checkSymmetric (int a[],int n)
{
    int i=0,j=n-1;
    while(i<=j)
            {
                        if(a[i]!=a[j]) return 0;
                                i++;
                                j--;
            }
    return 1;
}
```

Exercise 10.8

- Write a function that reverse the array content. Use this function in a program that asks user to enter a list of floatting numbers.

- Then reverse all these numbers without creating another array.

Solution

```
void reverse(float a[],int size)
{
  int i;
  float tmp;
  for (i=0; i<n/2; i++)
    {
        tmp=a[i]; a[i]=a[n-i-1];
        a[n-i-1]=tmp;

    }

}
```

Thank you for your attentions!

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

soict.hust.edu.vn/   fb.com/groups/soict