Mảng - Array

Duc-Minh VU @ ORLab - Phenikaa

Thực hành

- Đọc hiểu các đoạn code C có trong tài liệu (chương 3)
- Chuyển các đoạn code C sang code C++
- Tìm hiểu các hàm có sẵn trong C++ giúp viết code ngắn gọn
  - max/min/swap/sort/etc.
- Tìm hiểu array và vector trong STL C++
  - Bài 1: Vector - Khái niệm - Sử dụng thư viện chuẩn STL cho C/C++ (vncoder.vn)
  - Bài 2: Các hàm thường dùng trong Vector - Sử dụng thư viện chuẩn STL cho C/C++ (vncoder.vn)
  - vector - C++ Reference (cplusplus.com)
  - array - C++ Reference (cplusplus.com)
  - https://codelearn.io/learning/cau-truc-du-lieu-va-giai-thuat/1059946

Array

- Là một tập các phần tử cùng kiểu dữ liệu
- Các phần tử của mảng được lưu trữ ở trong các ô nhớ liên tiếp và mỗi phần tử có thể được tham chiếu bằng chỉ số là các số nguyên.

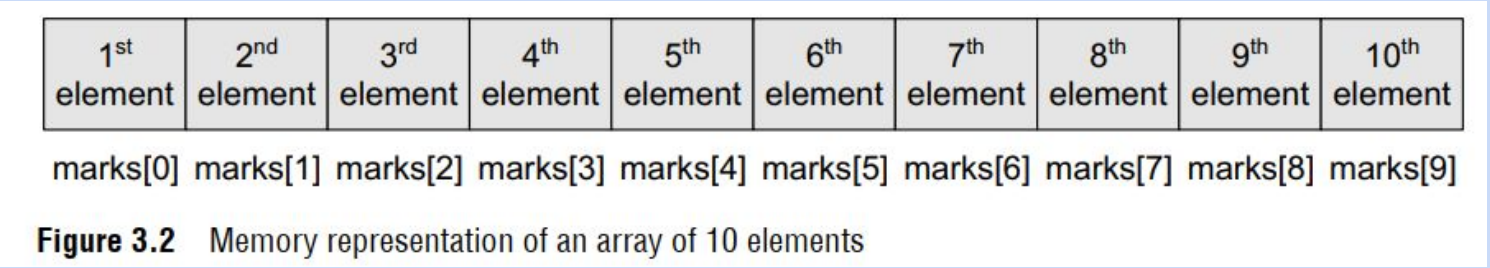


Figure 3.2 Memory representation of an array of 10 elements

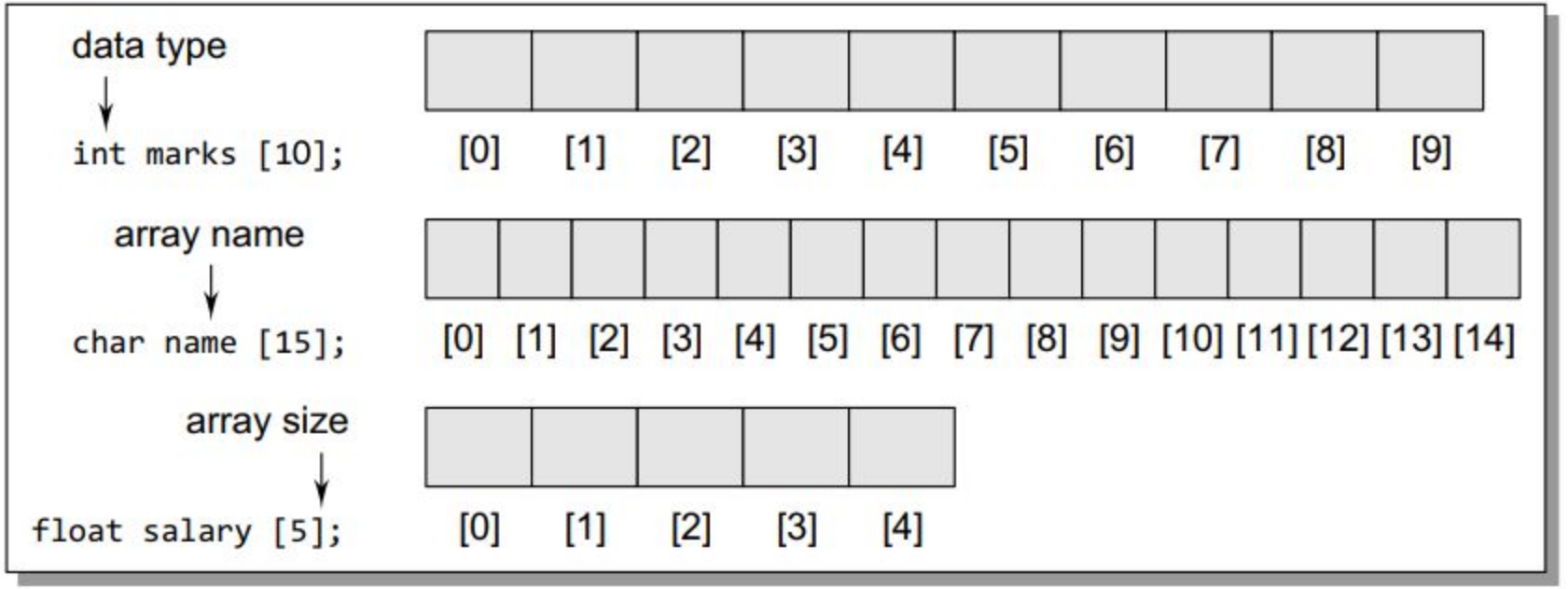


Figure 3.3 Declaring arrays of different data types and sizes

Gồm kiểu dữ liệu, tên mảng, và kích thước.

Khai báo mảng

Truy cập phần tử mảng

- Sử dụng toán tử []
- hoặc con trỏ

```
// Set each element of the array to -1
int i, marks[10];
for(i=0;i<10;i++)
        marks[i] = -1;
```

Figure 3.4 Code to initialize each element of the array to –1

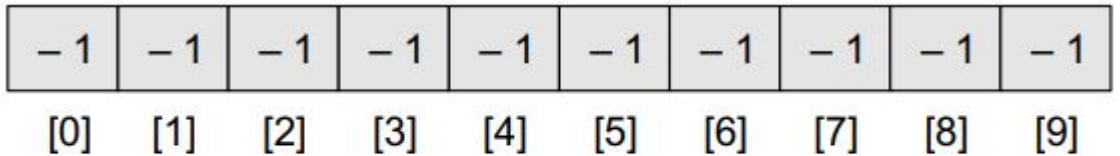


Figure 3.5 Array `marks` after executing the code given in Fig. 3.4

Example 3.1 Given an array `int marks[]={99,67,78,56,88,90,34,85}`, calculate the address of `marks[4]` if the `base address = 1000`.

Solution

99	67	78	56	**88**	90	34	85
marks[0]	marks[1]	marks[2]	marks[3]	**marks[4]**	marks[5]	marks[6]	marks[7]
1000	1002	1004	1006	**1008**	1010	1012	1014

We know that storing an integer value requires 2 bytes, therefore, its size is 2 bytes.

```
marks[4] = 1000 + 2(4 - 0)
         = 1000 + 2(4) = 1008
```

&a[i] = a + sizeof(a[0])*i // lý thuyết
&a[i] = a + i // máy tính

Tính địa chỉ phần tử của mảng

Example 3.2 Let `Age[5]` be an array of integers such that

`Age[0] = 2, Age[1] = 5, Age[2] = 3, Age[3] = 1, Age[4] = 7`

Show the memory representation of the array and calculate its length.

Solution

The memory representation of the array `Age[5]` is given as below.


```
Length = upper_bound - lower_bound + 1
```

Here, `lower_bound = 0, upper_bound = 4`

Therefore, `length = 4 - 0 + 1 = 5`

C: sizeof(a) / sizeof(a[0])
C++: size(a) // từ C++ 11

Tính độ dài mảng

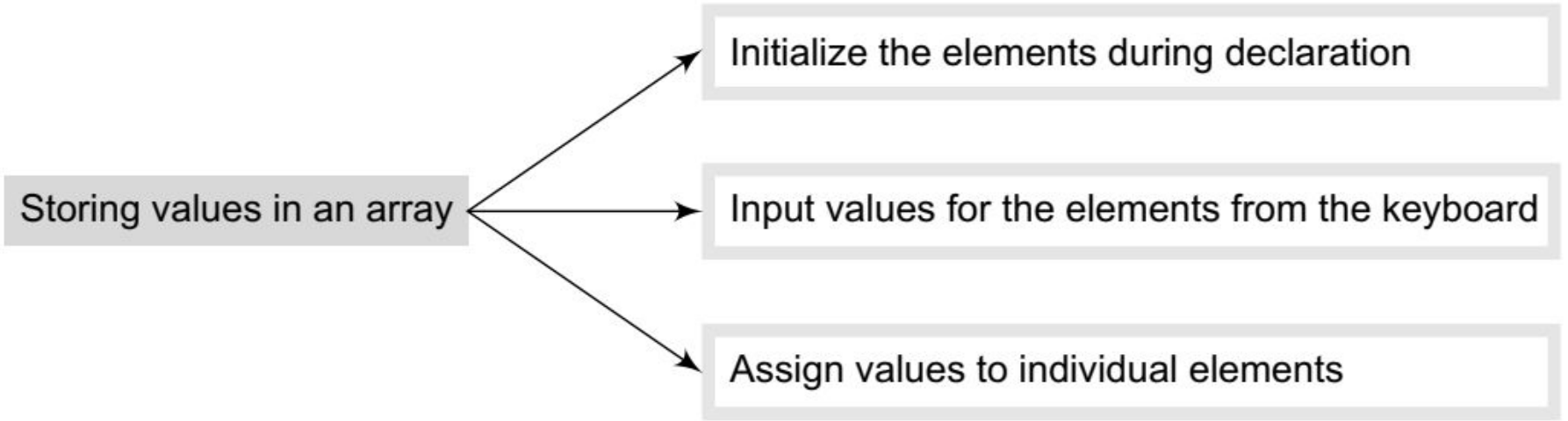


Figure 3.6 Storing values in an array

- Cập nhật giá trị trong khi khởi tạo
- Cập nhật qua bàn phím
- Cập nhật qua phép gán

Cập nhật giá trị mảng

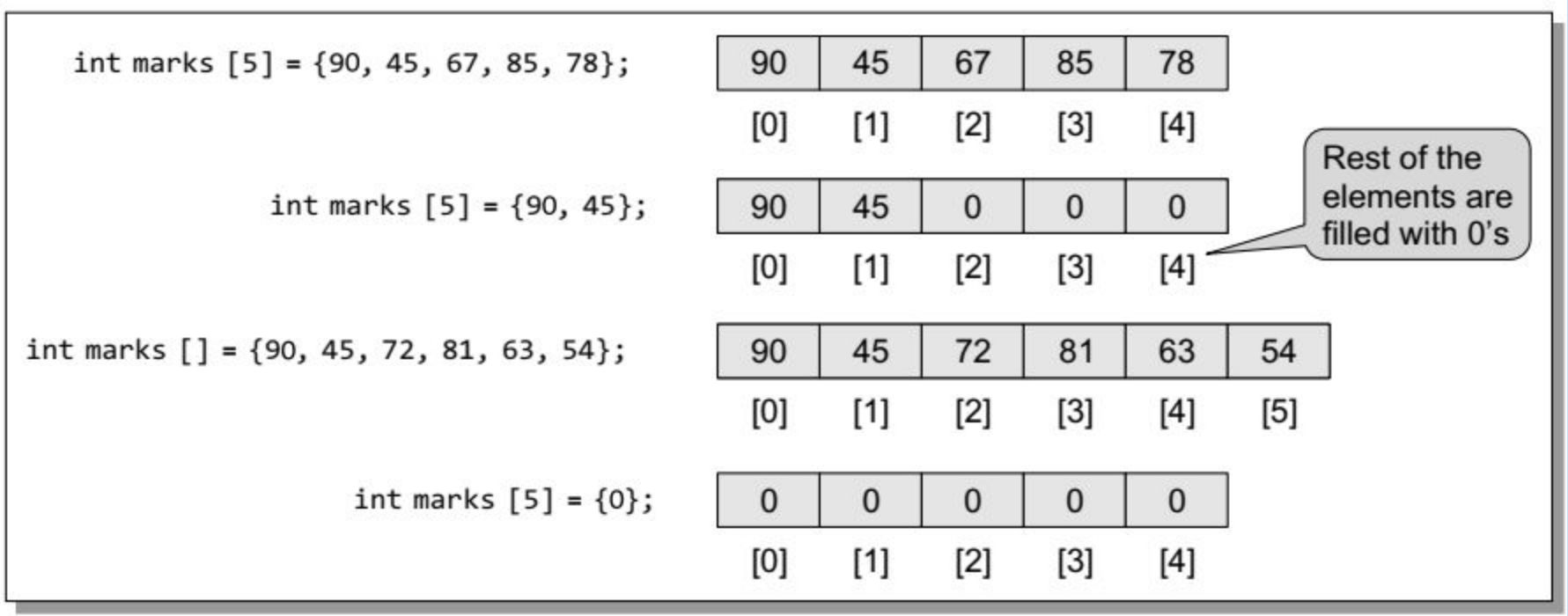


Figure 3.8 Initialization of array elements

Thiết lập giá trị qua khởi tạo bằng danh sách (từ C/C++11)

```
int i, marks[10];
for(i=0;i<10;i++)
        scanf("%d", &marks[i]);
```

Figure 3.9 Code for inputting each element of the array

C++: cin >> mark[i]

```
int i, arr1[10], arr2[10];
arr1[10] = {0,1,2,3,4,5,6,7,8,9};
for(i=0;i<10;i++)
        arr2[i] = arr1[i];
```

Figure 3.10 Code to copy an array at the individual element level

```
// Fill an array with even numbers
int i,arr[10];
for(i=0;i<10;i++)
        arr[i] = i*2;
```

Figure 3.11 Code for filling an array with even numbers

Thiết lập giá trị qua nhập từ bàn phím và qua phép gán

Các thao tác trên mảng

- Duyệt mảng
- Chèn phần tử vào mảng
- Tìm kiếm phần tử
- Xóa phần tử mảng
- Trộn hai mảng
- Sắp xếp mảng tăng dần/giảm dần

```
Step 1: [INITIALIZATION] SET I = lower_bound
Step 2: Repeat Steps 3 to 4 while I <= upper_bound
Step 3:       Apply Process to A[I]
Step 4:       SET  I = I + 1
        [END OF LOOP]
Step 5: EXIT
```

Figure 3.12 Algorithm for array traversal

- C: for/while/do while
- C++: C + for-based range loop / for_each
- Loops in C and C++ - GeeksforGeeks
- for_each loop in C++ - GeeksforGeeks
- Range-based for loop in C++ - GeeksforGeeks

Duyệt mảng

```cpp
// Illustration of range-for loop using CPP code
#include <bits/stdc++.h>
using namespace std;

int main()
{
    // Iterating over whole array
    std::vector<int> v = {0, 1, 2, 3, 4, 5};
    for (auto i : v) std::cout << i << ' ';
    std::cout << '\n';

    // the initializer may be a braced-init-list
    for (int n : {0, 1, 2, 3, 4, 5}) std::cout << n << ' ';
    std::cout << '\n';

    // Iterating over array
    int a[] = {0, 1, 2, 3, 4, 5};
    for (int n : a) std::cout << n << ' ';

    std::cout << '\n';

    // Printing string characters
    std::string str = "Hello World!";
    for (char c : str) std::cout << c << ' ';
    std::cout << '\n';
}
```

stdout

```
0 1 2 3 4 5
0 1 2 3 4 5
0 1 2 3 4 5
H e l l o  W o r l d !
```

for-based range loop C++ 11

```cpp
// C++ code to demonstrate the working
// of for_each with Exception

#include <iostream>
#include <algorithm>
using namespace std;

void print(int a)
{
    if (a & 1)
        cout << a * 2 << " ";
}

int main()
{
    // Initializing array
    int arr[] = {1, 5, 2, 4, 3};
    cout << "Using Array" << endl;

    for_each(arr, arr + 5, print);
    cout << endl;

    return 0;
}
```

stdout

```
Using Array
2 10 6
```

nằm trong thư viện algorithm

[for_each - C++ Reference (cplusplus.com)]

for_each loop C++

1. Write a program to read and display *n* numbers using an array.

```
#include <stdio.h>
#include <conio.h>
int main()
{
        int i, n, arr[20];
        clrscr();
        printf("\n Enter the number of elements in the array : ");
        scanf("%d", &n);
        for(i=0;i<n;i++)
        {
                printf("\n arr[%d] = ", i);
                scanf("%d",&arr[i]);
        }
        printf("\n The array elements are ");
        for(i=0;i<n;i++)
                printf("\t %d", arr[i]);
        return 0;
}
```

Độ phức tạp: O(n)

Chuyển sang code C++

2. Write a program to find the mean of *n* numbers using arrays.

```
#include <stdio.h>
#include <conio.h>
int main()
{
        int i, n, arr[20], sum =0;
        float mean = 0.0;
        clrscr();
        printf("\n Enter the number of elements in the array : ");
        scanf("%d", &n);
        for(i=0;i<n;i++)
        {
                printf("\n arr[%d] = ", i);
                scanf("%d",&arr[i]);
        }
        for(i=0;i<n;i++)
                sum += arr[i];
        mean = (float)sum/n;
        printf("\n The sum of the array elements = %d", sum);
        printf("\n The mean of the array elements = %.2f", mean);
        return 0;
}
```

Sử dụng cin/accumulate

Độ phức tạp: O(n)

accumulate - C++ Reference (cplusplus.com)

3. Write a program to print the position of the smallest number of *n* numbers using arrays.

```
#include <stdio.h>
#include <conio.h>
int main()
{
        int i, n, arr[20], small, pos;
        clrscr();
        printf("\n Enter the number of elements in the array : ");
        scanf("%d", &n);
        printf("\n Enter the elements : ");
        for(i=0;i<n;i++)
                scanf("%d",&arr[i]);
        small = arr[0]
        pos =0;
        for(i=1;i<n;i++)
        {
                if(arr[i]<small)
                {
                        small = arr[i];
                        pos = i;
                }
        }
        printf("\n The smallest element is : %d", small);
        printf("\n The position of the smallest element in the array is : %d", pos);
        return 0;
}
```

Độ phức tạp: O(n)

[std::min_element in C++ - GeeksforGeeks]

[max_element in C++ - GeeksforGeeks]

[y9MyUy - Online C++0x Compiler & Debugging Tool - Ideone.com]

Tìm số nhỏ nhất - dùng hàm min_element

6. Write a program to find whether the array of integers contains a duplicate number.

```
#include <stdio.h>
#include <conio.h>
int main()
{
        int array[10], i, n, j, flag =0;
        clrscr();
        printf("\n Enter the size of the array : ");
        scanf("%d", &n);
        for(i=0;i<n;i++)
        {
                printf("\n array[%d] = ", i);
                scanf("%d", &array[i]);
        }
        for(i=0;i<n;i++)
        {
                for(j=i+1;j<n;j++)
                {
                        if(array[i] == array[j] && i!=j)
                        {
                        flag =1;
                        printf("\n Duplicate numbers found at locations %d and %d", i, j)
                        }
                }
        }
        if(flag==0)
                printf("\n No Duplicates Found");
        return 0;
}
```

Độ phức tạp: O(n^2)

Kiểm tra có số nào trùng nhau hay không?

```
Step 1: Set upper_bound = upper_bound + 1
Step 2: Set A[upper_bound] = VAL
Step 3: EXIT
```

Figure 3.13 Algorithm to append a new element to an existing array

Chèn vào cuối mảng

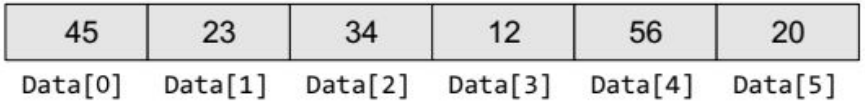
```
Step 1: [INITIALIZATION] SET I = N
Step 2: Repeat Steps 3 and 4 while I >= POS
Step 3:          SET A[I + 1] = A[I]
Step 4:          SET I = I - 1
        [END OF LOOP]
Step 5: SET N = N + 1
Step 6: SET A[POS] = VAL
Step 7: EXIT
```

Figure 3.14 Algorithm to insert an element in the middle of an array.

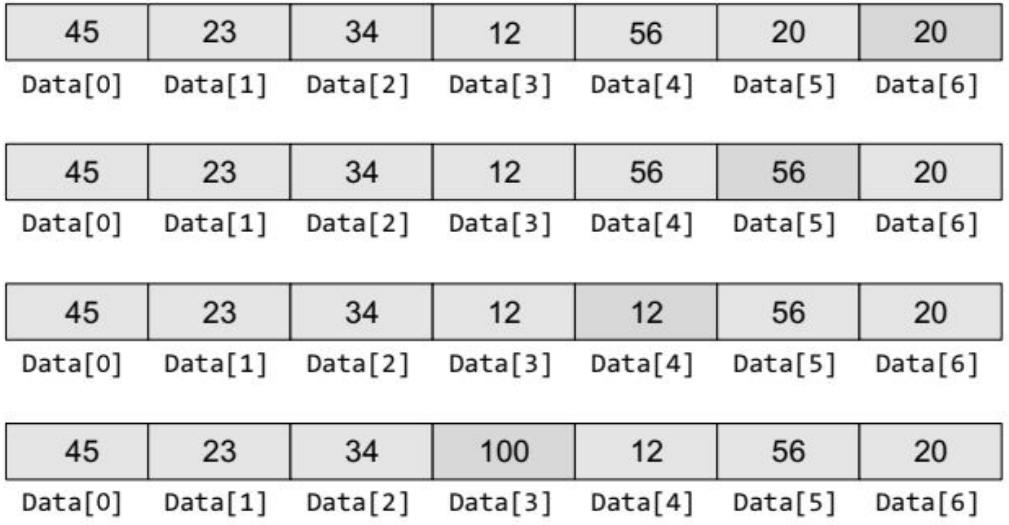
Chèn vào giữa mảng

Chèn phần tử vào mảng

Initial `Data[]` is given as below.



Calling `INSERT(Data, 6, 3, 100)` will lead to the following processing in the array:



Minh họa chèn vào giữa mảng

7. Write a program to insert a number at a given location in an array.

```
#include <stdio.h>
#include <conio.h>
int main()
{
        int i, n, num, pos, arr[10];
        clrscr();
        printf("\n Enter the number of elements in the array : ");
        scanf("%d", &n);
        for(i=0;i<n;i++)
        {
                printf("\n arr[%d] = ", i);
                scanf("%d", &arr[i]);
        }
        printf("\n Enter the number to be inserted : ");
        scanf("%d", &num);
        printf("\n Enter the position at which the number has to be added :");
        scanf("%d", &pos);
        for(i=n-1;i>=pos;i--)
                arr[i+1] = arr[i];
        arr[pos] = num;
        n = n+1;
        printf("\n The array after insertion of %d is : ", num);
        for(i=0;i<n;i++)
            printf("\n arr[%d] = %d", i, arr[i]);
        getch();
        return 0;
}
```

C++ có lớp vector, hỗ trợ thao tác chèn/xóa.

vector::push_back - C++ Reference (cplusplus.com) - chèn vào cuối vector

vector::insert - C++ Reference (cplusplus.com) - chèn vào giữa vector

Chèn phần tử vào giữa mảng

```cpp
#include <bits/stdc++.h>
using namespace std;

int main()
{
    int n, num, pos;
    vector<int> arr;

    cout << "Enter the number of elements in the array : ";
    cin >> n;
    for (int i = 0; i < n; i++)
    {
        int x;
        cout << "\n arr[" << i << "] = ";
        cin >> x;
        arr.push_back(x);
    }
    cout << "\n Enter the number to be inserted : "; cin >> num;
    cout << "\n Enter the position at which the number has to be added :"; cin >> pos;

    arr.insert(arr.begin() + pos, num);
    printf("\n The array after insertion of %d is : ", num);

    for (auto v : arr)
        cout << v << " ";
    return 0;
}
```

[zy366u - Online C++0x Compiler & Debugging Tool - Ideone.com]

Chèn phần tử vào mảng

```
Step 1: SET upper_bound = upper_bound - 1
Step 2: EXIT
```

Figure 3.15 Algorithm to delete the last element of an array

Xóa cuối mảng

```
Step 1: [INITIALIZATION] SET I = POS
Step 2: Repeat Steps 3 and 4 while I <= N - 1
Step 3:          SET A[I] = A[I + 1]
Step 4:          SET I = I + 1
        [END OF LOOP]
Step 5: SET N = N - 1
Step 6: EXIT
```

Figure 3.16 Algorithm to delete an element from the middle of an array

Xóa giữa mảng

Xóa phần tử mảng

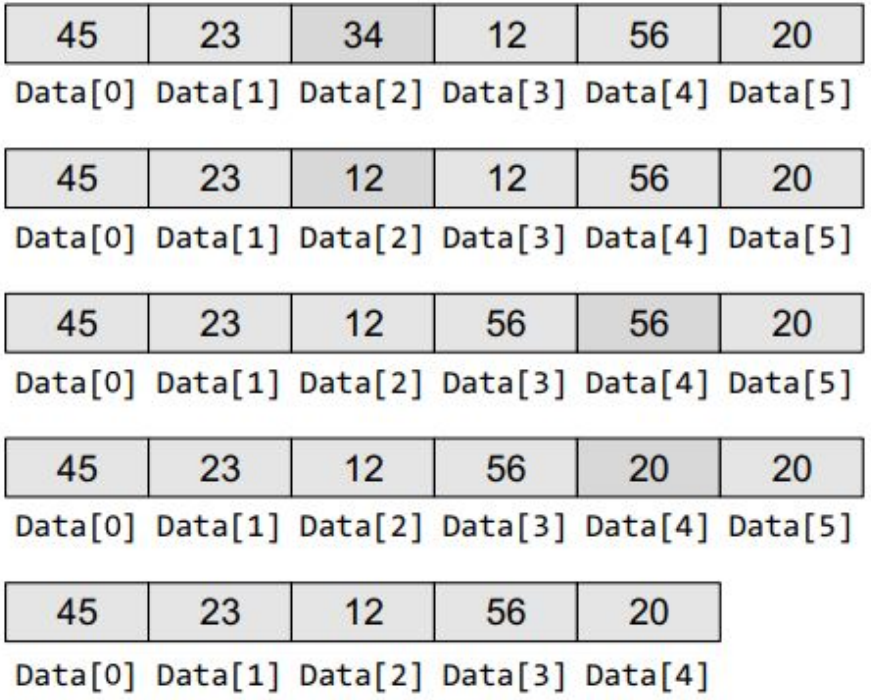


Figure 3.17 Deleting elements from an array

[vector::pop_back - C++ Reference (cplusplus.com)](#) - xóa phần tử cuối vector

[vector::erase - C++ Reference (cplusplus.com)](#)

- xóa phần tử ở giữa vector

https://www.cplusplus.com/reference/vector/vector/erase/ - xóa toàn bộ vector

Xóa phần tử khỏi mảng

```
Step 1: SET upper_bound = upper_bound - 1
Step 2: EXIT
```

Figure 3.15 Algorithm to delete the last element of an array

Xóa phần tử cuối mảng;

```
Step 1: [INITIALIZATION] SET I = POS
Step 2: Repeat Steps 3 and 4 while I <= N - 1
Step 3:          SET A[I] = A[I + 1]
Step 4:          SET I = I + 1
        [END OF LOOP]
Step 5: SET N = N - 1
Step 6: EXIT
```

Figure 3.16 Algorithm to delete an element from the middle of an array

Xóa phần tử giữa mảng

Xóa phần tử khỏi mảng

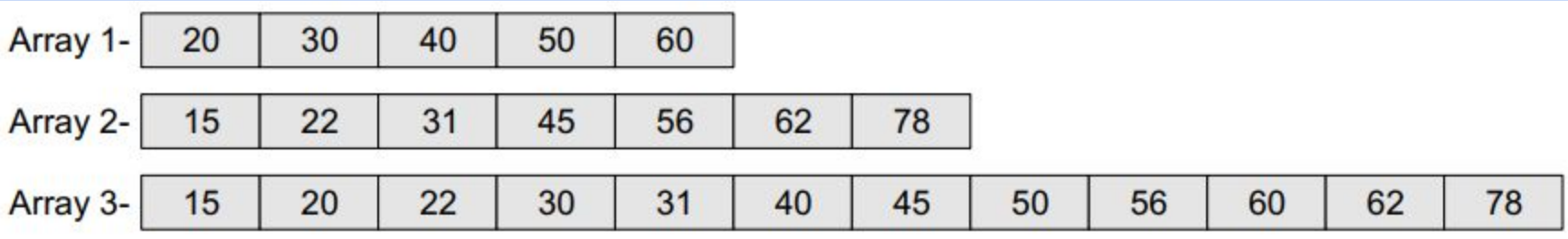


Figure 3.19 Merging of two sorted arrays

Phép toán quan trọng trong sắp xếp trộn
C++ có hàm merge hỗ trợ thao tác này
merge - C++ Reference (cplusplus.com)

PROGRAMMING EXAMPLE

12. Write a program to merge two sorted arrays.

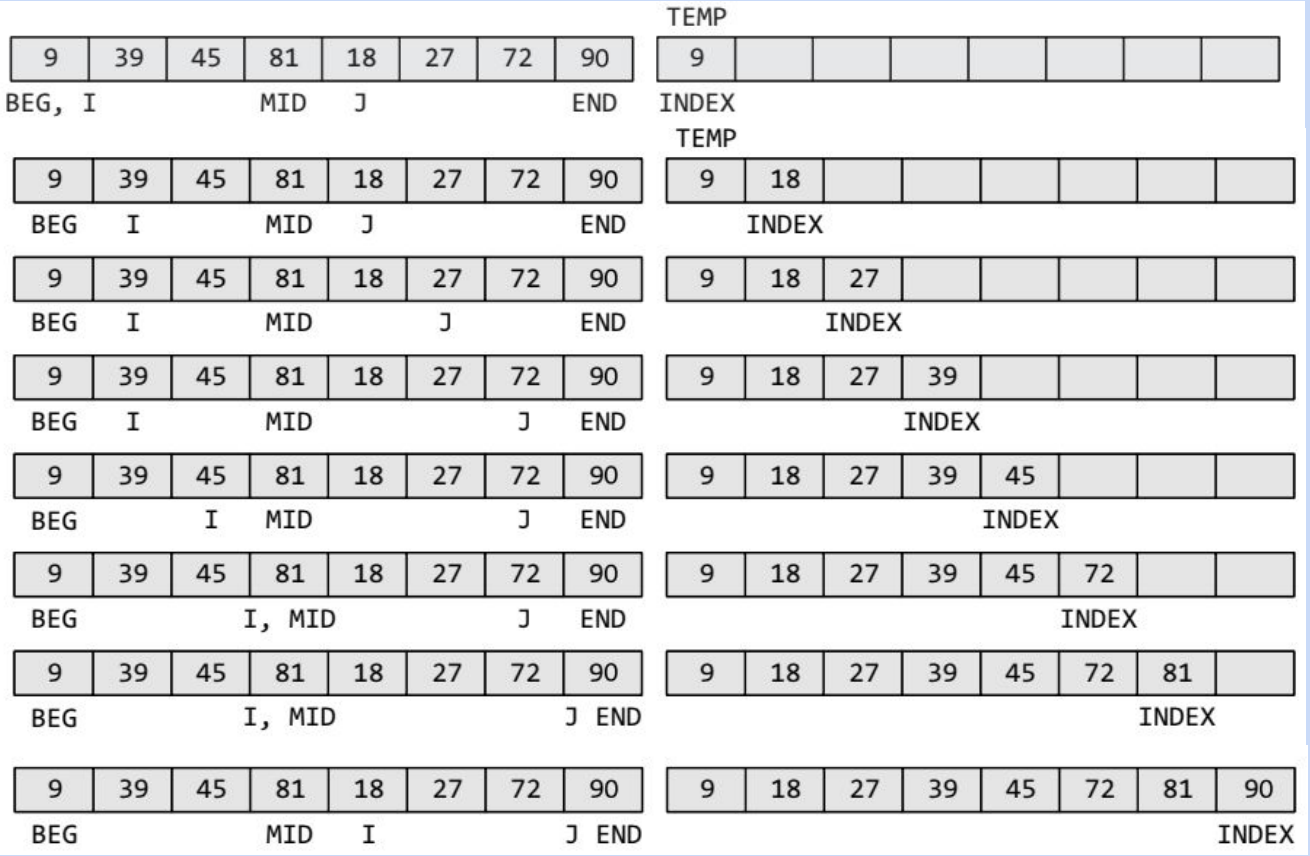
Trộn hai mảng tăng dần để thu được mảng tăng dần

```
MERGE (ARR, BEG, MID, END)

Step 1: [INITIALIZE] SET I = BEG, J = MID + 1, INDEX = 0
Step 2: Repeat while (I <= MID) AND (J<=END)
            IF ARR[I] < ARR[J]
                  SET TEMP[INDEX] = ARR[I]
                  SET I = I + 1
            ELSE
                  SET TEMP[INDEX] = ARR[J]
                  SET J = J + 1
            [END OF IF]
            SET INDEX = INDEX + 1
      [END OF LOOP]
Step 3: [Copy the remaining elements of right sub-array, if any]
            IF I > MID
               Repeat while J <= END
                  SET TEMP[INDEX] = ARR[J]
                  SET INDEX = INDEX + 1, SET J = J + 1
               [END OF LOOP]
      [Copy the remaining elements of left sub-array, if any]
            ELSE
               Repeat while I <= MID
                  SET TEMP[INDEX] = ARR[I]
                  SET INDEX = INDEX + 1, SET I = I + 1
               [END OF LOOP]
            [END OF IF]
Step 4: [Copy the contents of TEMP back to ARR] SET K=0
Step 5: Repeat while K < INDEX
               SET ARR[K] = TEMP[K]
               SET K = K + 1
      [END OF LOOP]
Step 6: END
```

Trộn hai dãy tăng dần thành một dãy tăng dần.

Idea: 1) so sánh phần tử hiện tại của hai dần; đưa phần tử nhỏ hơn vào dãy kết quả; 2) khi một trong hai dãy đã được đưa hết vào dãy kết quả, đưa toàn bộ các phần tử chưa được xét của dãy còn lại vào dãy kết quả.



9 < 18: đưa 9 vào kết quả
39 > 18: đưa 18 vào kết quả
39 > 27: đưa 27 vào kết quả
39 < 72: đưa 39 vào kết quả
…

Trộn mảng

```cpp
// merge algorithm example
#include <iostream>  // std::cout
#include <algorithm> // std::merge, std::sort
#include <vector>    // std::vector
using namespace std;

int main()
{
     int first[] = {5, 10, 15, 20, 25};
     int second[] = {50, 40, 30, 20, 10};
     int results[10];

     sort(first, first + 5);
     sort(second, second + 5);
     merge(first, first + 5, second, second + 5, results);

     cout << "The resulting array contains:";
     for (auto v : results)
          cout << v << " ";
     cout << '\n';

     return 0;
}
```

stdout

```
5 10 10 15 20 20 25 30 40 50
```

[ymYiki - Online C++0x Compiler & Debugging Tool - Ideone.com]

merge trong C++

```
21      /* Merge the temp arrays back into arr[l..r]*/
22      i = 0; // Initial index of first subarray
23      j = 0; // Initial index of second subarray
24      k = 0; // Initial index of merged subarray
25      while (i < n1 && j < n2)
26      {
27          if (L[i] <= R[j]) //
28              arr[k++] = L[i++];
29          else
30              arr[k++] = R[j++];
31      }
32
33      /* Copy the remaining elements of L[], if there are any */
34      while (i < n1) arr[k++] = L[i++];
35
36      /* Copy the remaining elements of R[], if there are any */
37      while (j < n2) arr[k++] = R[j++];
```

Thao tác trộn

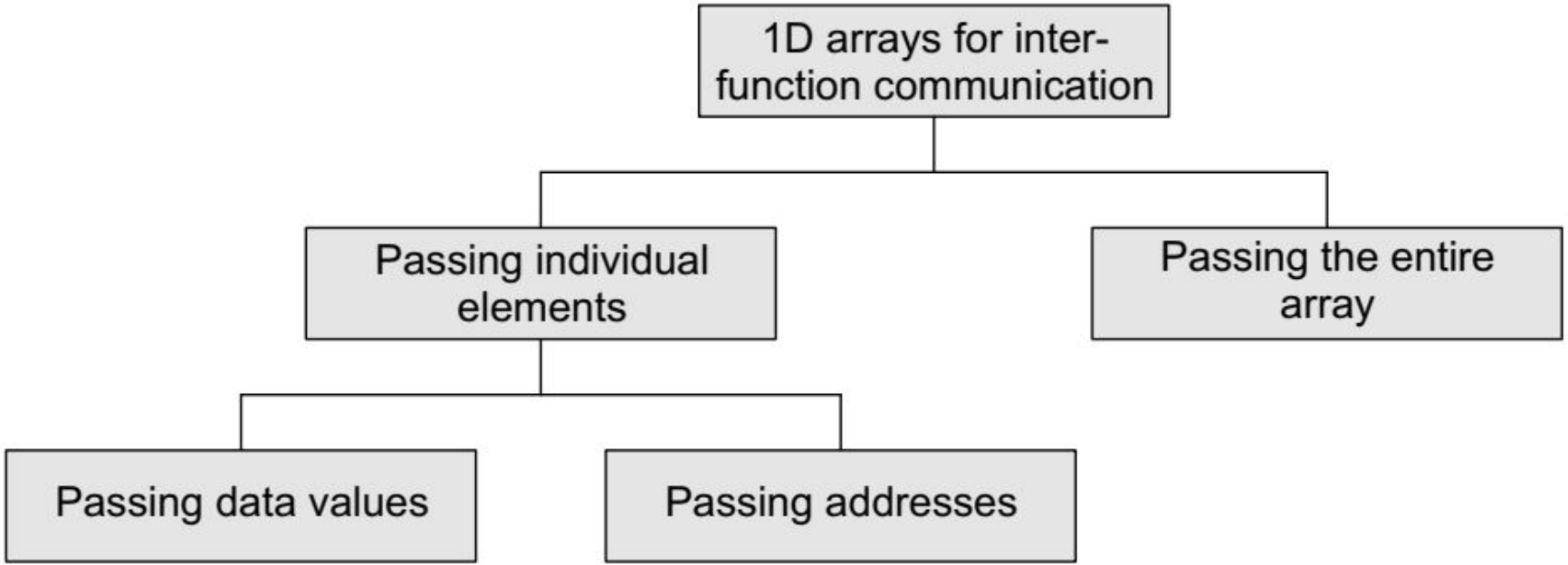


Figure 3.20 One dimensional arrays for inter-function communication

Truyền mảng cho hàm

Calling function

```
main()
{
        int arr[5] ={1, 2, 3, 4, 5};
        func(arr[3]);
}
```

Called function

```
void func(int num)
{
        printf("%d", num);
}
```

Figure 3.21(a) Passing values of individual array elements to a function

Làm việc trên bản sao của đối số được truyền vào

Truyền giá trị

Calling function	Called function

`main()`
`{`
`int arr[5] ={1, 2, 3, 4, 5};`
`func(&arr[3]);`
`}`
	`void func(int *num)`
	`{`
	`printf("%d", *num);`
	`}`

Figure 3.21(b) Passing addresses of individual array elements to a function

Làm việc trên bản sao của đối số được truyền vào - ở đây đối số là một địa chỉ

Truyền theo giá trị

Calling function

```
main()
{
        int arr[5] ={1, 2, 3, 4, 5};
        func(arr);
}
```

Called function

```
void func(int arr[5])
{
        int i;
        for(i=0;i<5;i++)
                printf("%d", arr[i]);
}
```

Figure 3.22 Passing entire array to a function

Truyền địa chỉ mảng theo giá trị.
Định nghĩa hàm func(int arr[5]) hay func(int arr[]) hay func(int* arr) là như nhau

Truyền theo giá trị

Con trỏ và mảng

- Tên mảng là con trỏ hằng (con trỏ trỏ tới 1 địa chỉ cố định).
-

```
int *ptr;
ptr = &arr[0];
```

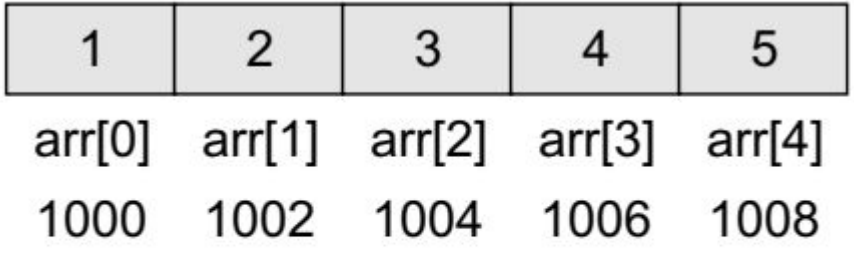


Figure 3.23 Memory representation of `arr[]`

Programming Tip

The name of an array is actually a pointer that points to the first element of the array.

th
cc

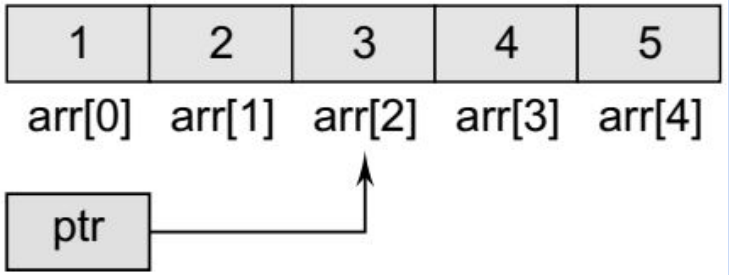


Figure 3.24 Pointer pointing to the third element of the array

```
main()
{
    int arr[]={1,2,3,4,5};
    printf("\n Address of array = %p %p %p", arr, &arr[0], &arr);
}
```

arr là con trỏ trỏ tới phần tử thứ nhất của mảng
&arr[0] là địa chỉ của phần tử thứ nhất của mảng
&arr là địa chỉ của toàn bộ mảng

Note arr[i], i[arr], *(arr+i), *(i+arr) gives the same value.

Con trỏ và mảng

```cpp
#include <iostream>

/* Driver code */
int main()
{
    int arr[] = {12, 11, 13, 5, 6, 7};
    std::cout << arr << " " << &arr[0] << " " << &arr << std::endl;
    std::cout << arr + 1 << " " << &arr[0] + 1 << " " << &arr + 1 << std::endl;
    return 0;
}
```

stdout

```
0x7ffec5a643a0 0x7ffec5a643a0 0x7ffec5a643a0
0x7ffec5a643a4 0x7ffec5a643a4 0x7ffec5a643b8
```

Con trỏ trỏ tới (toàn bộ mảng) và mảng các con trỏ

- int *p[10] là mảng 10 con trỏ kiểu int.
- int (*p)[10] là con trỏ trỏ tới mảng kiểu int 10 phần tử

```
int arr[]={1,2,3,4,5};
int *parr;
parr = arr;
```

```
int arr[2][2]={{1,2},{3,4}};
int (*parr)[2];
parr = arr;
```

```
int arr[2][2][2]={1,2,3,4,5,6,7,8};
int (*parr)[2][2];
parr = arr;
```