Arrays and Strings

Ho Dac Hung

Array Fundamentals

- In everyday life we commonly group similar objects into units. In computer languages we also need to group together data items of the same type. The most basic mechanism that accomplishes this in C++ is the array.
- Arrays can hold a few data items or tens of thousands. The data items grouped in an array can be simple types such as int or float, or they can be user-defined types such as structures and objects.

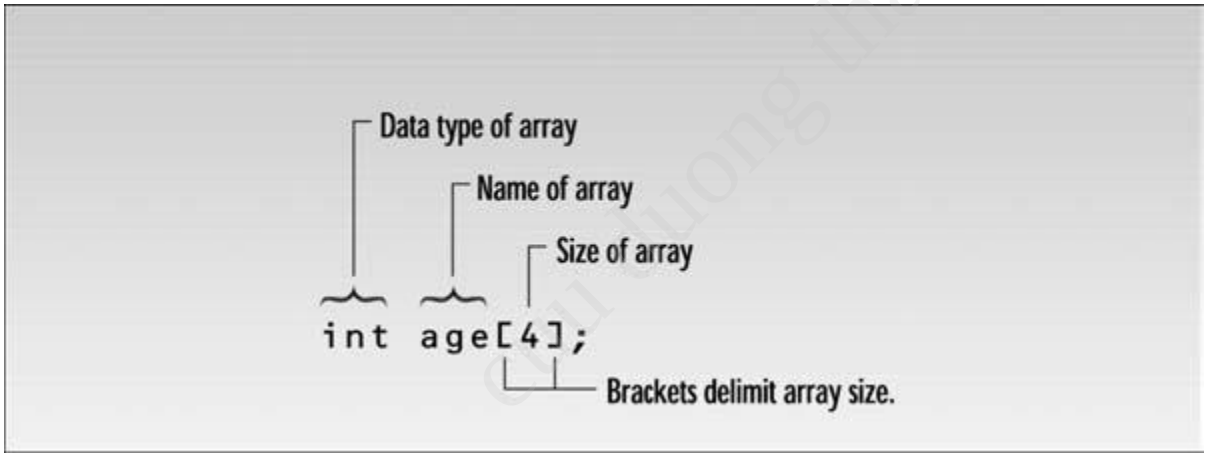
```
int main()
{
    int age[4];
    for(int j=0; j<4; j++) //get 4 ages
    {
        cout << “Enter an age: “;
        cin >> age[j]; //access array element
    }
    for(j=0; j<4; j++) //display 4 ages
        cout << “You entered “ << age[j] << endl;
    return 0;
}
```

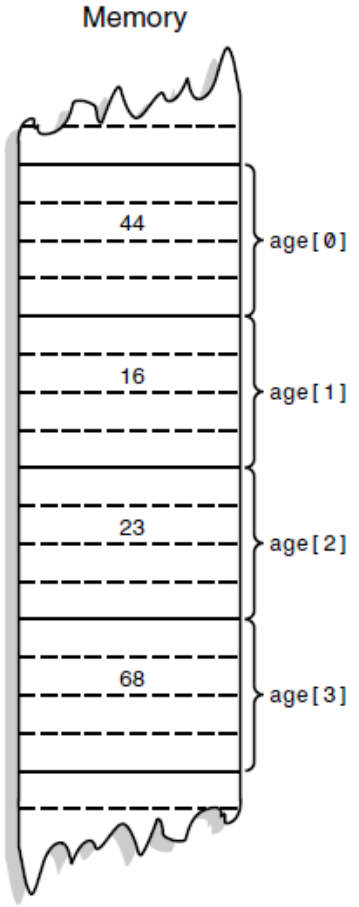
Array Fundamentals

- Like other variables in C++, an array must be defined before it can be used to store information. And, like other definitions, an array definition specifies a variable type and a name. But it includes another feature: a size. The size specifies how many data items the array will contain.

Array Fundamentals

- The items in an array are called elements. Notice that the first array element is numbered 0

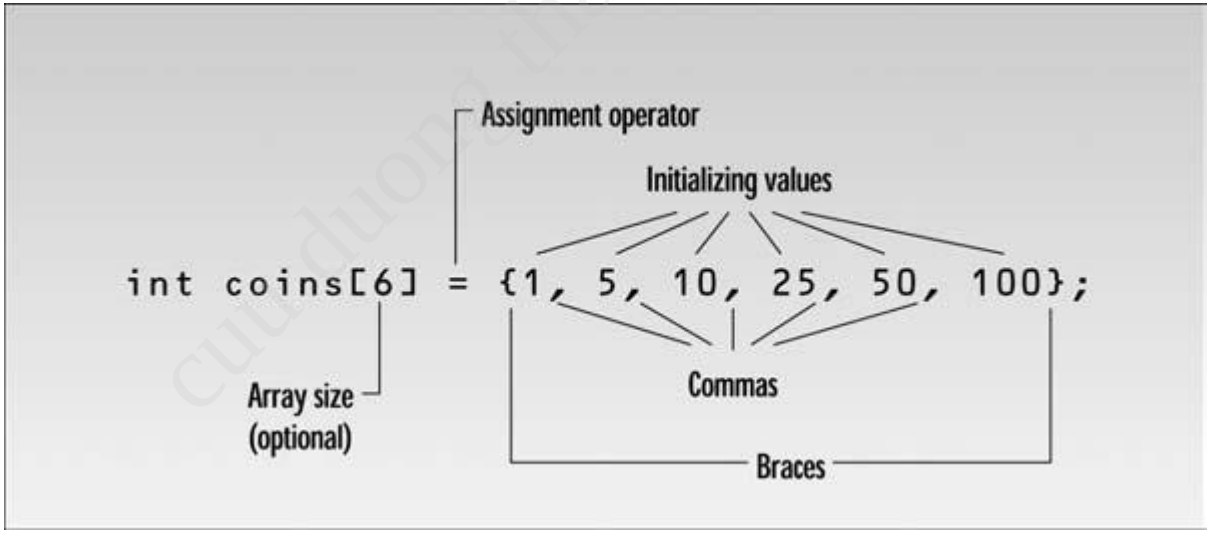




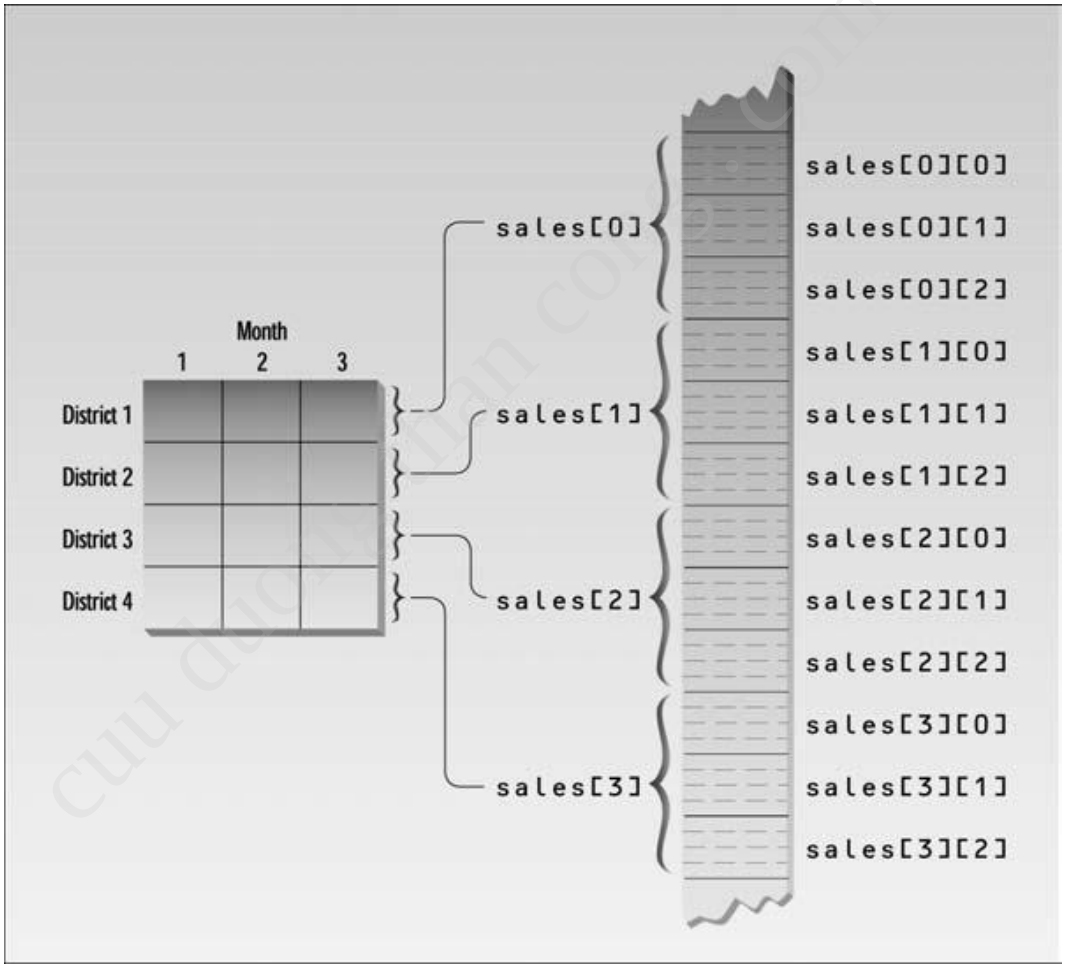
Initializing Arrays

- You can give values to each array element when the array is first defined.

```
int days_per_month[12] = { 31, 28, 31, 30,
                           31, 30, 31, 31, 30, 31, 30, 31 };
```



Multidimensional Arrays



```
const int DISTRICTS = 4;
const int MONTHS = 3;
int main()
{
    int d, m;
    double sales[DISTRICTS][MONTHS];
    cout << endl;
    for(d=0; d<DISTRICTS; d++)
        for(m=0; m<MONTHS; m++)
        {
            cout << “Enter sales for district “ << d+1;
            cout << “, month “ << m+1 << “: “;
            cin >> sales[d][m];
```

```
cout << “\n\n”;
cout << “ Month\n”;
cout << “ 1 2 3”;
for(d=0; d<DISTRICTS; d++)
{
    cout <<”\nDistrict “ << d+1;
    for(m=0; m<MONTHS; m++)
        cout << setiosflags(ios::fixed)
        <<setiosflags(ios::showpoint)
        << setprecision(2)<<setw(10)<< sales[d][m];
}
cout << endl;
return 0;
```

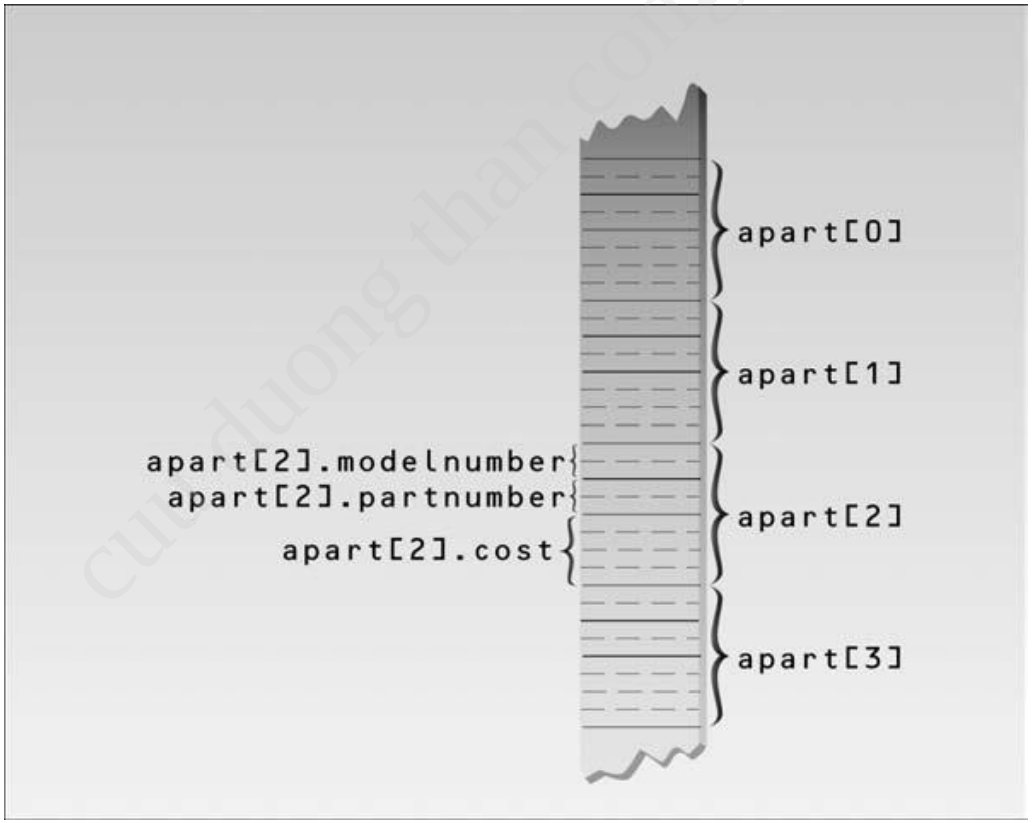
Passing Arrays to Functions

- Arrays can be used as arguments to functions.

```
void display( float[DISTRICTS][MONTHS] );
void display( float[][MONTHS] );
```

Arrays of Structures

- Arrays can contain structures as well as simple data types.



Arrays as Class Member Data

- Arrays can be used as data items in classes.

```
class Stack
{
    private:
        int st[MAX]; //stack: array of integers
        int top; //number of top of stack
    public:
        Stack() //constructor
        { top = 0; }
        void push(int var) //put number on stack
        { st[++top] = var; }
        int pop() //take number off stack
        { return st[top--]; }
};
```

Arrays of Objects

- We've seen how an object can contain an array. We can also reverse that situation and create an array of objects.

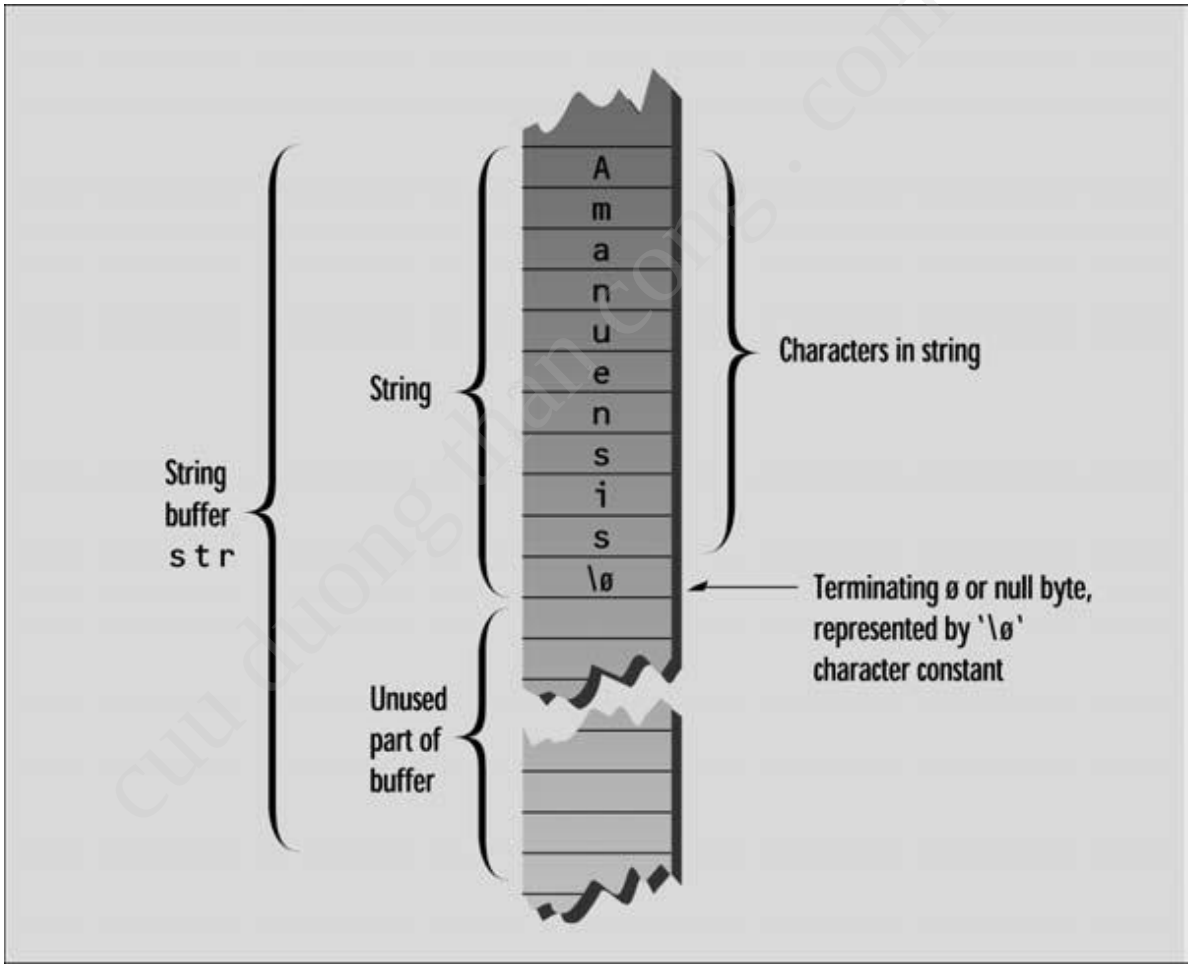
C-Strings

- There are two kinds of strings are commonly used in C++: C-strings and strings that are objects of the string class.

C-String Variables

```
int main()
{
    const int MAX = 80; //max characters in string
    char str[MAX];
    cout << “Enter a string: “;
    cin >> str; //put string in str
    //display string from str
    cout << “You entered: “ << str << endl;
    return 0;
}
```

C-String Variables



C-String Variables

```
int main()
{
const int MAX = 20; //max characters in string
char str[MAX]; //string variable str
cout << “\nEnter a string: “;
cin >> setw(MAX) >> str; //put string in str,
// no more than MAX chars
cout << “You entered: “ << str << endl;
return 0;
}
```

C-String Variables

```
int main()
{
    char str[] = “Farewell! thou art too dear for my
         possessing.”;
    cout << str << endl;
    return 0;
}
```

C-String Variables

```
int main()
{
    const int MAX = 80; //max characters in string
    char str[MAX]; //string variable str
    cout << “\nEnter a string: “;
    cin.get(str, MAX); //put string in str
    cout << “You entered: “ << str << endl;
    return 0;
}
```

C-String Variables

```
int main()
{
    cout << “\nEnter a string:\n”;
    cin.get(str, MAX, ‘$’); //terminate with $
    cout << “You entered:\n” << str << endl;
    return 0;
}
```

Copying a String

```
int main()
{ //initialized string
    char str1[] = “Hello world!”;
    const int MAX = 80; //size of str2 buffer
    char str2[MAX]; //empty string
    for(int j=0; j<strlen(str1); j++) //copy strlen characters
        str2[j] = str1[j]; // from str1 to str2
    str2[j] = ‘\0’; //insert NULL at end
    cout << str2 << endl; //display str2
    return 0;
}
```

Copying a String

```
int main()
{
    char str1[] = “Hello world!”
    const int MAX = 80; //size of str2 buffer
    char str2[MAX]; //empty string
    strcpy(str2, str1); //copy str1 to str2
    cout << str2 << endl; //display str2
    return 0;
}
```

The Standard C++ string Class

- Standard C++ includes a new class called string. This class improves on the traditional C-string in many ways.
- For one thing, you no longer need to worry about creating an array of the right size to hold string variables. The string class assumes all the responsibility for memory management.
- Also, the string class allows the use of overloaded operators, so you can concatenate string objects with the + operator.

```
int main()
{
    string s1("Man"); //initialize
    string s2 = "Beast"; //initialize
    string s3;
    s3 = s1; //assign
    cout << "s3 = " << s3 << endl;
    s3 = "Neither " + s1 + " nor "; //concatenate
    s3 += s2; //concatenate
    cout << "s3 = " << s3 << endl;
    s1.swap(s2); //swap s1 and s2
    cout << s1 << " nor " << s2 << endl;
    return 0;
}
```

```
int main()
{ //objects of string class
    string full_name, nickname;
    string greeting("Hello, ");
    cout << "Enter your full name: ";
    getline(cin, full_name); //reads embedded blanks
    cout << "Your full name is: " << full_name << endl;
    cout << "Enter your nickname: ";
    cin >> nickname; //input to string object
    greeting += nickname; //append name to greeting
    cout << greeting << endl; //output: "Hello, Jim"
    return 0;
}
```

```
int main()
{
    string s1 = “In Xanadu did Kubla Kahn a stately
        pleasure dome decree”;
    int n;
    n = s1.find(“Kubla”);
    cout << “Found Kubla at “ << n << endl;
    n = s1.find_first_of(“spde”);
    cout << “First of spde at “ << n << endl;
    n = s1.find_first_not_of(“aeiouAEIOU”);
    cout << “First consonant at “ << n << endl;
    return 0;
}
```

```
int main()
{
    string s1("Quick! Send for Count Graystone.");
    string s2("Lord");
    string s3("Don't ");
    s1.erase(0, 7); //remove "Quick! "
    s1.replace(9, 5, s2); //replace "Count" with "Lord"
    s1.replace(0, 1, "s"); //replace 'S' with 's'
    s1.insert(0, s3); //insert "Don't " at beginning
    s1.erase(s1.size()-1, 1); //remove '.'
    s1.append(3, '!'); //append "!!!"
    return 0;
}
```

```
int main()
{
string aName = “George”;
string userName;
cout << “Enter your first name: “;
cin >> userName;
if(userName==aName) //operator ==
cout << “Greetings, George\n”;
else if(userName < aName) //operator <
cout << “You come before George\n”;
else
cout << “You come after George\n”;
return 0;
}
```

```
int main()
{
    string aName = “George”;
    string userName;
    cout << “Enter your first name: “;
    cin >> userName;
    int n = userName.compare(0, 2, aName, 0, 2);
    cout << “The first two letters of your name “;
    if(n==0)
        cout << “match “;
    else if(n < 0)
        cout << “come before “;
    else
        cout << “come after “;
    cout << aName.substr(0, 2) << endl;
    return 0;
}
```

```
int main()
{
    char charray[80];
    string word;
    cout << “Enter a word: “;
    cin >> word;
    int wlen = word.length(); //length of string object
    cout << “One character at a time: “;
    for(int j=0; j<wlen; j++)
        cout << word.at(j); //exception if out-of-bounds
        // cout << word[j]; //no warning if out-of-bounds
    word.copy(charray, wlen, 0); //copy string object to array
    charray[wlen] = 0; //terminate with ‘\0’
    cout << “\nArray contains: “ << charray << endl;
    return 0;
```