C/C++ PROGRAMMING

# Sequential Search Array of Structured Records



## Structured Records

A structured record can contain multiple fields of the same or different data types. For example, a structured record can contain a character array for a customer's name, an integer for the account number, and a double for the balance on the account. The keyword **struct** is used in C and C++. The **struct** statement defines what a record will look like. Some languages do the same thing but call it a **Record**.

```
struct S_CUSTOMER {
    char name[40];
    int  acctNo;
    double balance;
};
```

# Typedef

A user defined data type can be created for the struct statement, and can even be combined when writing the struct statement. Example:

```
typedef  struct  S_CUSTOMER {
  char name[40];
  int  acctNo;
  double balance;
} customer;
```

```
customer Dan = {"Dan Mc", 55872, 357.83};
customer Sarah = {"Sarah Jones", 66987, 297.22};
```

# Search an Array of Structured Records

In this example, the program will ask the user to enter an abbreviation, search the **Abbrev** field of each record for a match, and then display the **Name** field if a match is found.

If a match was not found, a "Not Found" message will be displayed.

| | Abbrev | Name |
|---|---|---|
| 0 | AL | Alabama |
| 1 | AK | Alaska |
| 2 | AZ | Arizona |
| 3 | AR | Arkansas |
| 4 | CA | California |
| 5 | CO | Colorado |
| 6 | CT | Connecticut |
| 7 | DE | Delaware |
| 8 | FL | Florida |
| 9 | GA | Georgia |
| 10 | HI | Hawaii |
| 11 | ID | Idaho |
| 12 | IL | Illinois |
| 13 | IN | Indiana |
| 14 | IA | Iowa |
| 15 | KS | Kansas |
| 16 | KY | Kentucky |
| 17 | LA | Louisiana |
| 18 | ME | Maine |
| 19 | MD | Maryland |
| 20 | MA | Massachusetts |
| 21 | MI | Michigan |

# Sample Program – part 1/2

```
// Search_Struct_Sequential.cpp : Defines the entry point for the console application.
//
#include "stdafx.h"     // only for Microsoft Visual C++
#include <iostream>
using namespace std;

typedef struct S_STATE_LIST {
        char StateName[15];
        char Abbreviation[3];
} States;

States StateList [] = {
        {"Alabama", "AL"}, {"Alaska", "AK"}, {"Arizona", "AZ"}, {"Arkansas", "AR"},
        {"California", "CA"}, {"Colorado", "CO"}, {"Connecticut", "CN"}, {"Delaware", "DE"},
        {"Florida", "FL"}, {"Georgia", "GA"}, {"Hawaii", "HI"}, {"Idaho", "ID"},
        {"Illinois", "IL"}, {"Indiana", "IN"}, {"Iowa", "IA"}, {"Kansas", "KS"},
        {"Kentucky", "KY"}, {"Louisiana", "LA"}, {"Maine", "ME"}, {"Maryland", "MD"},
        {"Massachusetts", "MA"}, {"Michigan", "MI"}, {"Minnesota", "MN"},
        {"Mississippi", "MS"}, {"Missouri", "MO"}, {"Montana", "MT"}, {"Nebraska", "NE"},
        {"Nevada", "NV"}, {"New Hampshire", "NH"}, {"New Jersey", "NJ"}, {"New Mexico", "NM"},
        {"New York", "NY"}, {"North Carolina", "NC"}, {"North Dakota", "ND"}, {"Ohio", "OH"},
        {"Oklahoma", "OK"}, {"Oregon", "OR"}, {"Pennsylvania", "PA"}, {"Rhode Island", "RI"},
        {"South Carolina", "SC"}, {"South Dakota", "SD"}, {"Tennessee", "TN"}, {"Texas", "TX"},
        {"Utah", "UT"}, {"Vermont", "VT"}, {"Virginia", "VA"}, {"Washington", "WA"},
        {"West Virginia", "WV"}, {"Wisconsin", "WI"}, {"Wyoming", "WY"}
};
```

# Sample Program – part 2/2

```
int main(int argc, char* argv[])
{
    int i;
    int length = 50;

    char Selection[10];
    cout << "Enter the state abbreviation: ";
    cin  >> Selection;

    // Sequential Search
    for (i=0; i<length; i++)
    {
        if ( _stricmp (Selection, StateList[i].Abbreviation) == 0)
        {
            cout << StateList[i].StateName << endl;
            break;
        }
    }
    if ( i == length )
        cout << "Not Found" << endl;

    return 0;
}
```