Структури

Базово програмиране
Лекция 8
Милена Карова
кат. КНТ ТУ-Варна

Въведение в структурите

- Определение за структура -Структурата е една или няколко променливи от различни типове, групирани заедно.
- Обединяване на стойности в едно цяло
- Обръщане към структурата като към една единствена променлива
- Възможност за инициализация, въвеждане и редактиране на отделни части от нея (полета)
- Използва се служебната дума structure или struct

Пример за структура

```
struct CDAccount
{
    double balance;
    double interest_rate;
    int term;//months until maturity
};
```

- Структурата обикновено се обявява глобално или в хедърен файл
- Поставянето на фигурални скоби и ; е задължително
- След като веднъж структурата е дефинирана, то към нея може да се обръщаме като към всеки друг дефиниран тип.

```
CDAccount my_account, your_account;
```

Пример за структура

```
//Program to demonstrate the CDAccount structure type.
#include <iostream>
using namespace std;

//Structure for a bank certificate of deposit:
struct CDAccount
{
    double balance;
    double interest_rate;
    int term;//months until maturity
};


void get_data(CDAccount& the_account);
//Postcondition: the_account.balance and the_account.interest_rate
//have been given values that the user entered at the keyboard.


int main( )
{
    CDAccount account;
    get_data(account);

    double rate_fraction, interest;
    rate_fraction = account.interest_rate/100.0;
    interest = account.balance*rate_fraction*(account.term/12.0);
    account.balance = account.balance + interest;

    cout.setf(ios::fixed);
    cout.setf(ios::showpoint);
    cout.precision(2);
    cout << "When your CD matures in "
         << account.term << " months,\n"
         << "it will have a balance of $"
         << account.balance << endl;
    return 0;
}

//Uses iostream:
void get_data(CDAccount& the_account)
{
    cout << "Enter account balance: $";
    cin >> the_account.balance;
    cout << "Enter account interest rate: ";
    cin >> the_account.interest_rate;
    cout << "Enter the number of months until maturity\n"
         << "(must be 12 or fewer months): ";
    cin >> the_account.term;
}
```

Sample Dialogue

```
Enter account balance: $100.00
Enter account interest rate: 10.0
Enter the number of months until maturity
(must be 12 or fewer months): 6
When your CD matures in 6 months,
it will have a balance of $105.00
```

Този пример структурата има 3 полета: balance, interest_rate и term

Действия:

```
account.balance
account.interest_rate
account.term
```

```
account.balance = 1000.00;
account.interest_rate = 4.7;
account.term = 11;
```

```
account.balance = account.balance + interest;
```

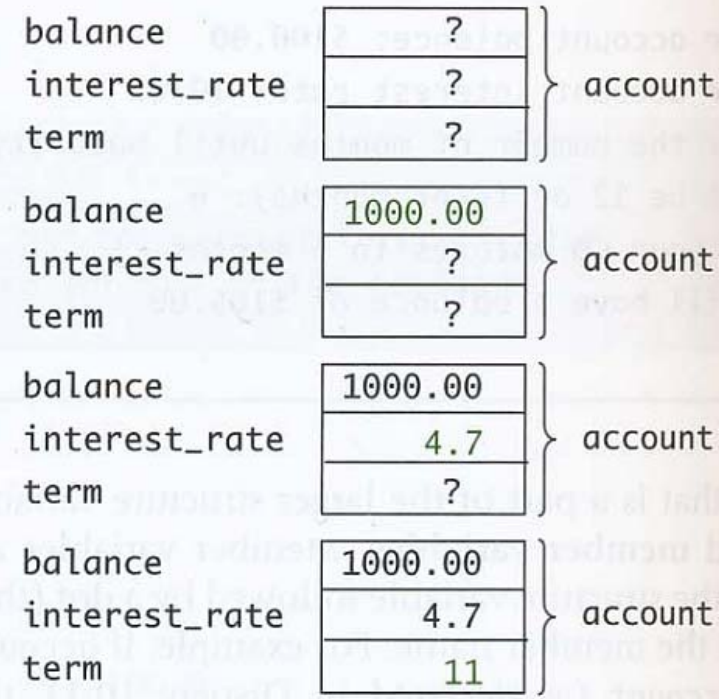
Разположение на данните в паметта

```
struct CDAccount
{
    double balance;
    double interest_rate;
    int term;//months until maturity
};
int main( )
{
   CDAccount account;
      ...

   account.balance = 1000.00;

   account.interest_rate = 4.7;

   account.term = 11;
```



Операции с променливи от тип структура

- Оператор точка (dot operator)

- Две структури могат да имат еднакви полета

```
struct FertilizerStock
{
    double quantity;
    double nitrogen_content;
};
```

```
struct CropYield
{
    int quantity;
    double size;
};
```

- Ако двете променливи apples и oranges са от една и съща структура то операцията на присвояване е абсолютно коректна

```
apples = oranges;
```

```
apples.quantity = oranges.quantity;
apples.size = oranges.size;
```

Променливи от тип структура

- Варианти на обявяване на променливи от тип структура

```
struct WeatherData
{
    double temperature;
    double wind_velocity;
} data_point1, data_point2;
```

- Обърнете внимание на позицията на ;
- Пример за оператор .

The Dot Operator

The **dot operator** is used to specify a member variable of a structure variable.

SYNTAX

Dot operator

```
Structure_Variable_Name.Member_Variable_Name
```

EXAMPLES

```
struct StudentRecord
{
    int student_number;
    char grade;
};

int main( )
{
    StudentRecord your_record;
    your_record.student_number = 2001;
    your_record.grade = 'A';
```

Структура и функции

- Елементите на структурата получават стойности от аргументите на функцията

```
CDAccount shrink_wrap(double the_balance,
                          double the_rate, int the_term)
{
    CDAccount temp;
    temp.balance = the_balance;
    temp.interest_rate = the_rate;
    temp.term = the_term;
    return temp;
}
```

- Структурата може да бъде и тип на връщаната от функцията стойност
- Извикване на функцията

```
CDAccount new_account;
new_account = shrink_wrap(10000.00, 5.1, 11);
```

Йерархични структури

```
struct Date
{
    int month;
    int day;
    int year;
};

struct PersonInfo
{
    double height;//in inches
    int weight;//in pounds
    Date birthday;
};
```

```
PersonInfo person1;
```

```
cout << person1.birthday.year;
```

Инициализиране на структури

```
struct Date
{
    int month;
    int day;
    int year;
};
```

```
Date due_date = {12, 31, 2004};
```

Примерна структура

```
struct person{
 char idn[6];
 float rate;
int hours;
double zaplata;
};
void vavejdane(struct person per[])
 {  cout<<endl<<“Въведете броя на сътрудниците“;
 cin>>n;
for (i=0;i<n;i++)
{ cout<<endl<<“Въведете данни за сътрудник номер“<<i+1<<endl;
cout<<endl<<“Идентификационен номер:”;
cin>>per[i].idn;
 cout<<endl<<“По часово заплащане:“;
cin>>per[i].rate;
cout<<endl<<“Отработени часове“;
cin>>per[i].hours;
per[i].zaplata=per[i]rate*per[i].hours;
}
```