

Структури

Калин Георгиев

26 февруари 2014 г.

“Пакетиране” на стойности

```
double distance (double x1, double y1, double x2, double y2)
{
    return sqrt ((x1-x2)*(x1-x2) - (y1-y2)*(y1-y2));
}
```

“Пакетиране” на стойности

```
/*??????*/ western (double x1, double y1, double x2, double y2)
{
    if (x1 < x2)
        return /* (x1,y1) */;

    return /* (x2,y2) */;
}
```

Структури

```
struct Point
{
    double x; //field x
    double y; //field y
}
```

- Дефиниране на променливи

```
double a;
int x,y;
Point p1, p2;
```

- Достъп до полета

```
p1.x = 10;
cout << p1.x;
p1.x = p2.x + 5;
```

- Връщане като резултат

```
Point western (Point p1, Point p2)
{
    if (p1.x < p2.x)
        return p1;
    return p2;
}
```

Пример

```
Point western (Point p1, Point p2)
{
    if (p1.x < p2.x)
        return p1;
    return p2;
}

int main ()
{
    Point p1,p2;
    cin << p1.x << p1.y << p2.x << p2.y;

    Point p3 = western (p1,p2);
    cout << "The western point is ("
         << p3.x
         << ", "
         << p3.y
         << ")" << endl;

    //cout << p3 ???
}
```

Пример: Рационални числа

```
struct Rational
{
    double nom, denom;
};
```

$$\frac{a_{nom}}{a_{denom}} + \frac{b_{nom}}{b_{denom}} = \frac{a_{nom} * b_{denom} + b_{nom} * a_{denom}}{a_{denom} * b_{denom}}$$

```
Rational sum (Rational a, Rational b)
{
    Rational result;
    result.nom = a.nom*b.denom + b.nom*a.denom;
    result.denom = a.denom * b.denom;
    return result;
}

Rational multiply (Rational a, Rational b)
{
    Rational result;
    result.nom = a.nom*b.nom;
    result.denom = a.denom*b.denom;
    return result;
}

void print (Rational a)
{
    cout << a.nom << "/" << a.denom;
}
```

Пример: Рационални числа

$$a * b + c$$

```
double an, ad, bn, db, cn, cd;  
//...  
cout << an*bn*cd + cn*ad*bd;  
      << "/"  
      << ad*bd*cd;
```

- Алтернативно:

```
Rational a,b,c;  
//...  
print (sum (multiply (a,b) , c));
```

По-сложни примери

```
struct Date
{
    int day, month, year;
};
struct Person
{
    char name[100];
    Date birthdate;
};
```

```
void readPerson (Person& p)
{
    cout << "Please enter name:";
    cin.getline (p.name,99);
    cout << "Please enter day, month, "
         << "and year:";
    cin >> p.birthdate.day
        >> p.birthdate.month
        >> p.birthdate.year;
}
void printPerson (Person p)
{
    cout << "Name:" << p.name
         << " birthdate:"
         << p.birthdate.day << "/"
         << p.birthdate.month << "/"
         << p.birthdate.year << endl;
}
```


По-сложни примери

```
struct Date
{
    int day, month, year;
};
struct Person
{
    char name[100];
    Date birthdate;
};
```

```
void readPerson (Person& p)
{
    cout << "Please enter name: ";
    cin.getline (p.name,99);
    cout << "Please enter day, month, "
         << "and year: ";
    cin >> p.birthdate.day
        >> p.birthdate.month
        >> p.birthdate.year;
}
void printPerson (Person p)
{
    cout << "Name: " << p.name
         << " birthdate: "
         << p.birthdate.day << "/"
         << p.birthdate.month << "/"
         << p.birthdate.year << endl;
}
```

Помощна функция

```
Date earlier (Date d1, Date d2)
{
    if (d1.year < d2.year) return d1;
    if (d1.year == d2.year &&
        d1.month < d2.month) return d1;
    if (d1.year == d2.year &&
        d1.month == d2.month &&
        d1.day < d2.day) return d1;

    return d2;
}
```

Масив от структури

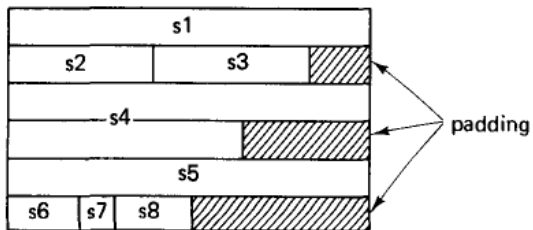
```
Person findYoungest (Person people[], int n)
{
    int index = 0;
    for (int i = 1; i < n; i++)
        if (earlier (people[i].birthdate, people[index].birthdate))
            index = i;
    return people[index];
}
```

Група от хора

```
Person people [10];  
int i;  
  
for (i=0; i<10; i++)  
    readPerson (people[i]);  
  
printPerson (findYoungest (people,10));
```

Представяне в паметта

Представяне в паметта



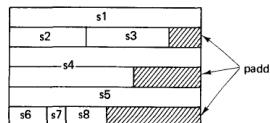
Фигура : Подравняване (padding)[1]

Представяне в паметта

```
struct S {Ta a; Tb b; Tc c;};  
S x;
```

- НЕ МОЖЕМ да разчитаме, че:

```
sizeof (S) == sizeof (Ta) + sizeof (Tb) + sizeof (Tc)  
(long)&x.b == (long)&x + sizeof (Ta);
```



Указатели и функции

Указатели

```
double *pb = &x.b; //double*
*pb = 10;
cout << *pb << x.b;
```

```
S arr[10];
pb = &arr[3].b;
*pb = 10;
cout << *pb << arr[3].b;
```

```
S* ps = &arr[3];
ps->b = 15;
cout << ps->b
      << (*ps).b
      << arr[3].b;
```

```
struct S
{
    int a;
    double b;
    char c;
};
S x;
```

Функции

```
void f (S z)
{
    cout << z.b;
    z.b = 10;
    cout << z.b;}

```

```
void g (S& z)
{cout << z.b; z.b = 20;}

```

```
void h (S* z)
{z->b = 30;}

```

```
S i (S z)
{cout << z.b; z.b = 40; return z;}

```

```
int main ()
{
    S x;
    x.b = 0;

    f(x); cout << x.b;
    g (x); cout << x.b;
    h (&x); cout << x.b;

    cout << i(x).b;
    cout << x.b;
}

```

Библиография



Niklaus Wirth. *“Algorithms + Data Structures = Programs”*, Prentice-Hall Series in Automatic Computation, 1976