

Функции от високо ниво II: Map.Reduce

Калин Георгиев

2 април 2020 г.

Map.Reduce

Задача: еднотипна промяна на всеки елемент на масив

```
void increase (int arr[], int arrsize)
{
    for (int i = 0; i < arrsize; i++)
        arr[i] = arr[i]+1;
}

void multiply (int arr[], int arrsize)
{
    for (int i = 0; i < arrsize; i++)
        arr[i] = arr[i]*2;
}

void increaseEvens (int arr[], int arrsize)
{
    for (int i = 0; i < arrsize; i++)
        if (arr[i] %2 == 0)
            arr[i] = arr[i] + 1;
}
```

Map

$$\text{operation} : T \rightarrow T$$

- еднотипна обработка на всеки от елементите на масив

```
template <typename T>
using opFn = T (*) (T);

template <typename T>
void map (T arr[], int arrsize, opFn<T> op)
{
    for (int i = 0; i < arrsize; i++)
        arr[i] = op(arr[i]);
}
```

Пример: добавяне на единица

```
int plusOne (int x)
{return x+1;}
```

```
int multTwo (int x)
{return x*2;}
```

```
int main ()
{
    int arr[] = {1,2,3};
    map<int> (arr,3,plusOne);
    map<int> (arr,3,multTwo);

    printArray<int> (arr,3);
}
```

```
template <typename T>
using opFn = T (*) (T);

template <typename T>
void map (T arr[], int arrsize, opFn<T> op)
{
    for (int i = 0; i < arrsize; i++)
        arr[i] = op(arr[i]);
}
```

Пример: добавяне на единица само на четните елементи

```
int evenPlusOne (int x)
{
    if (x%2 == 0)
        return x+1;
    return x;
}
```

```
int main ()
{
    int arr[] = {1,2,3};
    map<int> (arr,3,evenPlusOne);

    printArray (arr,3);
}
```

```
template <typename T>
using opFn = T (*) (T);

template <typename T>
void map (T arr[], int arrsize, opFn<T> op)
{
    for (int i = 0; i < arrsize; i++)
        arr[i] = op(arr[i]);
}
```

Задача: намиране на сума, произведение, брой и пр.

```
int sum (int arr[], int arrsize)
{
    int result = 0;
    for (int i = 0; i < arrsize; i++)
        result = result + arr[i];
    return result;
}

int prod (int arr[], int arrsize)
{
    int result = 1;
    for (int i = 0; i < arrsize; i++)
        result = result * arr[i];
    return result;
}

int countEvens (int arr[], int arrsize)
{
    int result = 0;
    for (int i = 0; i < arrsize; i++)
        if (arr[i] % 2 == 0)
            result = result + 1;
    return result;
}
```

Reduce

$$\text{combinator} : R \times E \rightarrow R$$

- Сумиране (“акумулиране”, “обединяване”) на всички елементи в един резултат

```
template <typename ResT, typename ElemT>
using combineFn = ResT (*) (ResT, ElemT);
```

```
template <typename ResT, typename ElemT>
ResT reduce (ElemT arr[],
             int arrsize,
             combineFn<ResT, ElemT> f,
             ResT null_val)
{
    ResT result = null_val;

    for (int i = 0; i < arrsize; i++)
        result = f (result, arr[i]);

    return result;
}
```


Пример: Събиране и умножение

```
int sum (int accumulated, int x)
{return accumulated + x;}
```

```
int prod (int accumulated, int x)
{return accumulated * x;}
```

```
int main ()
{
    int arr[] = {1,2,3};
    cout << reduce<int,int> (arr,3,sum,0);
    cout << reduce<int,int> (arr,3,prod,1);
}
```

```
template <typename ResT, typename ElemT>
using combineFn = ResT (*) (ResT, ElemT);

template <typename ResT,typename ElemT>
ResT reduce (ElemT arr[],
             int arrsize,
             combineFn<ResT,ElemT> f,
             ResT null_val)
{
    ResT result = null_val;

    for (int i = 1; i < arrsize; i++)
        result = f (result,arr[i]);

    return result;
}
```

Пример: Събиране само на четните числа

```

int sumEvens (int accumulated, int x)
{
    if (x % 2 == 0)
        return accumulated + x;
    return accumulated;
}

int main ()
{
    int arr[] = {1,2,3};
    cout << reduce<int,int> (arr,3,sumEvens,0);
}

template <typename ResT, typename ElemT>
using combineFn = ResT (*) (ResT, ElemT);

template <typename ResT,typename ElemT>
ResT reduce (ElemT arr[],
             int arrsize,
             combineFn<ResT,ElemT> f,
             ResT null_val)
{
    ResT result = null_val;

    for (int i = 1; i < arrsize; i++)
        result = f (result,arr[i]);

    return result;
}

```

Пример: Проверка дали има четни числа

```
bool isEven (bool accumulated, int x)
{
    if (x % 2 == 0)
        return true;
    return accumulated;
}
```

```
int main ()
{
    int arr[] = {1,2,3};
    cout << reduce<bool,int> (arr,3,isEven,false);
}
```

```
template <typename ResT, typename ElemT>
using combineFn = ResT (*) (ResT, ElemT);
```

```
template <typename ResT,typename ElemT>
ResT reduce (ElemT arr[],
             int arrsize,
             combineFn<ResT,ElemT> f,
             ResT null_val)
{
    ResT result = null_val;

    for (int i = 1; i < arrsize; i++)
        result = f (result,arr[i]);

    return result;
}
```

Пример: Брой срещания на символ

```
int countLs (int accumulated, char x)
{
    if (x == 'l')
        return accumulated + 1;
    return accumulated;
}
```

```
int main ()
{
    cout << reduce<int, char> ("Hello_World!", 12, countLs, 0);
}
```

```
template <typename ResT, typename ElemT>
using combineFn = ResT (*) (ResT, ElemT);
```

```
template <typename ResT, typename ElemT>
ResT reduce (ElemT arr[],
             int arrsize,
             combineFn<ResT, ElemT> f,
             ResT null_val)
{
    ResT result = null_val;

    for (int i = 1; i < arrsize; i++)
        result = f (result, arr[i]);

    return result;
}
```

Благодаря за вниманието!