



Data Structures and Algorithms

Структуры данных. Массивы



Массив

Массив - структура данных, хранящая набор значений (элементы массива), идентифицируемых по индексу или набору индексов. В качестве индексов используются целые числа из определенного диапазона. Особенностью массива является константная сложность получения элемента массива по индексу.

Набор операций предоставляемых массивом:

- Получение значения по индексу
- Установка значения по индексу

Размер массива — количество элементов в нем.

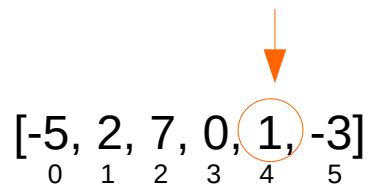
Размерность массива — минимальное количество индексов необходимое для однозначной адресации элемента массива.

Форма или структура массива — сведения о количестве размерностей и размере массива по каждой из размерностей. Может быть представлена одномерным массивом.



Размерность и форма массива

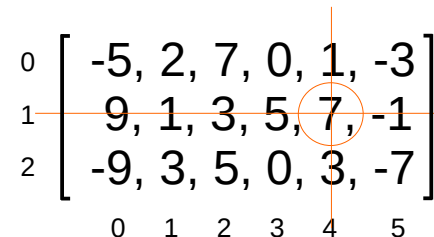
$[-5, 2, 7, 0, 1, -3]$
0 1 2 3 4 5



Одномерный (нужен один индекс).

Форма = [6]

$\begin{matrix} 0 & \left[\begin{array}{cccccc} -5, & 2, & 7, & 0, & 1, & -3 \end{array} \right] \\ 1 & \left[\begin{array}{cccccc} 9, & 1, & 3, & 5, & 7, & -1 \end{array} \right] \\ 2 & \left[\begin{array}{cccccc} -9, & 3, & 5, & 0, & 3, & -7 \end{array} \right] \end{matrix}$
0 1 2 3 4 5



Двумерный (нужно два индекса).

Форма = [3,6]



Гомогенные и гетерогенные массивы

Гомогенные массивы — массивы хранящие значения одного типа.

Гетерогенным называется массив, в разные элементы которого могут быть непосредственно записаны значения, относящиеся к различным типам данных. Массив, хранящий указатели на значения различных типов, не является гетерогенным, так как собственно хранящиеся в массиве данные относятся к единственному типу — типу «указатель».



Статические и динамические массивы

Статические массивы — массивы размер которых остается постоянным все время существования массива.

Динамические массивы — массивы размер которых может быть изменен во время существования массива. Наиболее частая реализация заключается в создании статического массива, и если в нем уже нет места под новый элемент, то создание нового статического массива большего размера копирование данных из первого и удаление первого массива.

Массивы переменной длины — массивы размер которых не известен на момент компиляции. Размер такого массива можно указать с помощью значения переменной или математическим выражением. Массивы переменной длины обычно к динамическим массивам не относят.



Python

Реализация в Python



Реализация массивов в Python

Поддержка массивов в Python на уровне стандартной библиотеки реализована только для небольшого списка типов данных. Это символ, целое число и вещественное число. Для остальных типов массивы не поддерживаются и нужно использовать списки или сторонние пакеты расширения (NumPy и т. д.). Для работы требуется импортировать модуль `array`. Поддерживается создание массива на основе строки, списка (целые и вещественные числа) и последовательности байт. При создании нужно указать тип данных, хранимых в массиве. Массивы гомогенные, динамические. Индексация начинается с нуля.



Таблица соответствий типов данных массива

Код типа	Тип в языке C	Тип в Python	Размер в байтах
b	signed char	int	1
B	unsigned char	int	1
u	wchar_t	Unicode character	2
h	signed short	int	2
H	unsigned short	int	2
i	signed int	int	2
I	unsigned int	int	2
l	signed long	int	4
L	unsigned long	int	4
q	signed long long	int	8
Q	unsigned long long	int	8
f	float	float	4
d	double	float	8



Методы поддерживаемые массивами для добавления, вставки и удаления элементов

Структура данных	Метод для создания массива на ее основе
append(x)	Вставка элемента в конец
insert(i, x)	Вставка элемента на i индекс
pop([i])	Удаление с получением элемента по индексу. Если индекс не указан то последний элемент.
remove(x)	Удаление элемента из массива
reverse()	Реверс массива



Пример использования массивов в Python

```
import array

arr = array.array('l', [0, -2, 5])
arr.append(6)
print(arr)

s = 0

for i in range(len(arr)):
    s += arr[i]

print(s)
```



Java

Реализация массивов в Java



Реализация массивов в Java

В Java поддерживаются как одномерные так и многомерные статические гомогенные массивы переменной длины. Массивы могут хранить данные любого типа который поддерживается. Индексация начинается с нуля.

Ограничения накладываемые на массивы:

- Максимальная размерность 255
- Максимальное количество элементов в одном измерении $2^{31}-1$

В Java массивы являются объектами. Поэтому кроме данных, хранят еще и вспомогательные данные. Одним из важных является поле **length** хранящие размер массива.



Создания массива в Java

Для создания массива и инициализации массива в Java используется следующий синтаксис.

```
Type [] name = new Type [size];
```

Type — тип данных хранимых в массиве. size — значение типа int указывает на размер создаваемого массива. Количество квадратных скобок обозначает размерность массива.

Если элементы массива заданы явно, то можно использовать следующий синтаксис для одномерного массива.

```
Type [] name = new Type[]{e1,e2,e3....};
```

e1,e2,e3 — элементы хранимые в массиве



Пример работы с одномерным массивом

```
int[] arr = new int[10];  
  
for (int i = 0; i < arr.length; i++) {  
    arr[i] = (int) (Math.random() * 10);  
}  
  
System.out.println(Arrays.toString(arr));
```



Пример работы с двумерным массивом

```
double[][] arr2 = new double[4][3];

for (int i = 0; i < arr2.length; i++) {
    for (int j = 0; j < arr2[i].length; j++) {
        arr2[i][j] = i * j;
    }
}
```

Довольно важным моментом является то, что индекс строки идет первым, а потом индекс столбца. Многомерные массивы не гарантируют, что будут выделены смежные области памяти для данных каждой размерности.



Fortran

Реализация массивов в Fortran

Поддержка массивов в Fortran

В Fortran поддерживаются как одномерные так и многомерные статические гомогенные массивы как фиксированной так и переменной длины. Массивы могут хранить данные любого типа. Индексация начинается с произвольно заданного целого индекса (по умолчанию с единицы).

Ограничения накладываемые на массивы:

- Максимальная размерность 15
- Максимальное количество элементов $2^{63}-1$. Зависит от компилятора и разрядности ос.

Массивы в Fortran не являются объектами. Поэтому для получения информации о форме, размере, минимальном и максимальном индексе нужно вызывать вспомогательные функции.

Пример работы с одномерным массивом (фиксированная длина)

```
program sample1
implicit none
integer(8),dimension(10)::arr
integer(8)::i
arr = 0

do i = 1, size(arr, dim = 1)
    write(*,*) arr(i)
end do

end program sample1
```

Внимание в Fortran корректность индекса не проверяется. Поэтому, в случае не внимательности можно выйти за границы массива, и получить доступ к некорректным данным или получить ошибку сегментации.

Пример работы с одномерным массивом (переменная длина)

```
program sample1
implicit none
integer(8),dimension(:),allocatable::arr
integer(8)::i, array_size

write(*,*) 'Input array size'
read(*,*) array_size

allocate(arr(array_size))

arr = 0

do i = 1, size(arr, dim = 1)
    write(*,*) arr(i)
end do

deallocate(arr)

end program sample1
```

Пример работы с двумерным массивом

```
integer(8), dimension(4,5)::arr
integer(8)::i,j
arr = 0

do i = 1, size(arr, dim = 2)
    do j = 1, size(arr, dim = 1)
        write(*,*) arr(j,i)
    end do
    write(*,*)
end do
```

Стандартом предусмотрено что память под двумерный массивы выделялась именно по столбцам, поэтому для увеличения быстродействия старайтесь описывать циклы, что бы быстрее менялись номера строк. Выполните полный проход по первому столбцу, потом по второму и т.д.

Некоторые функции для работы с массивами

Функция	Описание
<code>size(arr,[dim])</code>	Размер массива по указанному измерению. По умолчанию <code>dim = 1</code>
<code>lbound(arr,[dim])</code>	Вернет минимальный индекс по указанному измерению. По умолчанию <code>dim = 1</code>
<code>ubound(arr,[dim])</code>	Вернет максимальный индекс по указанному измерению. По умолчанию <code>dim = 1</code>
<code>shape(arr)</code>	Вернет форму массива

Пример применения некоторых встроенных функций

```
program sample4
implicit none
integer, dimension(0:10)::arr

write(*,*) size(arr)
write(*,*) lbound(arr,dim = 1)
write(*,*) ubound(arr,dim = 1)
write(*,*) shape(arr)

end program sample4
```



Список литературы

- 1) <https://docs.python.org/3/library/array.html>
- 2) <https://docs.oracle.com/javase/specs/jvms/se8/html/jvms-4.html#jvms-4.4.1>
- 3) <https://fortran-lang.org/>