

# Масиви и низове

# Логическо описание

## Масивът

- е съставен тип данни
- представя крайни редици от елементи
- всички елементи са от един и същи тип
- позволява произволен достъп до всеки негов елемент по номер (индекс)

# Дефиниция на масив

<тип> <идентификатор> [[<константа>]]  
[ = { <константа> {, <константа> } } ] ;

Примери:

- `bool b[10];`
- `double x[3] = { 0.5, 1.5, 2.5 }, y = 3.8;`
- `int a[] = { 3 + 2, 2 * 4 }; ↔ int a[2] = { 5, 8 };`
- `float f[4] = { 2.3, 4.5 }; ↔ float f[4] = { 2.3, 4.5, 0, 0 };`

# Физическо представяне

a

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]	a[7]	a[8]	a[9]	a[10]
------	------	------	------	------	------	------	------	------	------	-------

# Операции за работа с масиви

- Достъп до елемент по индекс:  $a[i]$ 
  - $x = a[2];$
  - $a[i] = 7;$  (**lvalue!**)
  - **Внимание: няма проверка за коректност!**
- Няма присвояване ( ~~$a = b$~~ )
- Сравнението не проверява всички елементи! ( $a == a$  връща винаги true)
- Няма операции за вход и изход



# Задачи за масиви

- Въвеждане на масив от числа
- Извеждане на масив от числа
- Намиране на сума на масив от числа
- Търсене на число в масив
- Проверка за монотонно нарастване
- Проверка за множество
- Сортиране на масив
- Сливане на два подредени масива

# Низове

- Последователност от символи, завършваща със символа '\0' (с код 0)
- Реализират се чрез масиви от символи

```
char word[] = { 'H', 'e', 'l', 'l', 'o', '\0' };
```

```
char word[6] = { 'H', 'e', 'l', 'l', 'o' };
```

```
char word[100] = "Hello";
```

**Грешно:** `char word[5] = "Hello";`

**Правилно:** `char word[6] = "Hello";`

# Операции за работа с низове

- Вход (>>, `cin.getline(<низ>)`) и изход (<<)
- Индексиране ([])
- Няма присвояване! ~~(a = b)~~
- Няма сравнение! ~~(a == b)~~
- Дължина на низ  
`strlen(<низ>)` — връща броя символи, без `\0`
- Копиране на низ  
`strcpy(<буфер>, <низ>)` — връща <буфер>



# Сравнение на низове

- `strcmp(<низ1>, <низ2>)` — сравнява два низа лексикографски (речникова наредба)
  - ◇ -1, ако <низ<sub>1</sub>> е преди <низ<sub>2</sub>>
  - ◇ 1, ако <низ<sub>1</sub>> е след <низ<sub>2</sub>>
  - ◇ 0, ако <низ<sub>1</sub>> и <низ<sub>2</sub>> съвпадат

Интуиция: знакът на израза  $\langle \text{низ}_1 \rangle - \langle \text{низ}_2 \rangle$

- `strcmp(s1, s2) == -strcmp(s2, s1)`

# Операции за работа с низове

- Конкатениране (слепване) на низове  
`strcat(<низ1>, <низ2>)` — записва <низ<sub>2</sub>> в края на <низ<sub>1</sub>>, връща <низ<sub>1</sub>>
- Търсене на символ в низ  
`strchr(<низ>, <символ>)` — връща суфикса на <низ> от първото срещане на <символ> нататък, или `false`, ако не се среща
- Търсене на подниз в низ  
`strstr(<низ>, <подниз>)` — връща суфикса на <низ> от първото срещане на <подниз> нататък, или `false`, ако не се среща

# Проблеми при работа с низове

- Излизане извън буфера (buffer overflow)

```
char a[10] = "Hello, world!";  
char b[] = "Hello,", c[] = " world!";  
strcat(b, c); strcpy(b, c);
```

- Нетерминирани низове

```
char a[5] = { 'H', 'e', 'l', 'l', 'o' };  
cout << strlen(a);  
char b[10];  
strcpy(b, a);
```

# Ограничени операции

- `strncpy(<буфер>, <низ>, n)`  
Копира първите  $n$  символа на `<низ>` в `<буфер>`, допълвайки с `'\0'` при нужда. Връща `<буфер>`
- `strncat(<низ1>, <низ2>, n)` — конкатенира първите  $n$  символа на `<низ2>` след `<низ1>`, завършвайки с `'\0'`
- `strncmp(<низ1>, <низ2>, n)` — сравнява първите  $n$  символа на `<низ1>` с `<низ2>`



# Задачи за низове

- Проверка за палиндром
- Преброяване на думи в низ
- Пресмятане на израз



# Многомерни масиви

- Масив от елементи, които са масиви
- `<тип> <идентификатор> [<константа>]`  
`{ [<константа> ] } = { <константа> { , <константа> } }`

Примери:

```
int a[2][3] = { { 1, 2, 3 }, {4, 5, 6}};
```

```
double b[5][6] = { 0.1, 0.2, 0.3, 0.4 };
```

```
int c[4][5] = { { 1, 2}, {3, 4, 5, 6}, {7, 8, 9}, {10} };
```

```
float f[][2][3] = { {{1.2, 2.3, 3.4}, {4.5, 5.6, 6.7}},  
                    {{7.8, 8.9, 9.1}, {1.2, 2.3, 3.4}},  
                    {{5.6}, {6.7, 7.8}}};
```

# Физическо представяне

a											
a[0]						a[1]					
a[0][0]			a[0][1]			a[1][0]			a[1][1]		
a[0][0][0]	a[0][0][1]	a[0][0][2]	a[0][1][0]	a[0][1][1]	a[0][1][2]	a[1][0][0]	a[1][0][1]	a[1][0][2]	a[1][1][0]	a[1][1][1]	a[1][1][2]

# Задачи за многомерни масиви

- Въвеждане и извеждане на матрица
- Транспониране на матрица
- Суми по стълбове
- Редовете, в които се среща  $x$
- Произведение от минимума по стълбове
- Обхождане по диагонал
- Шахматно сливане на две матрици