C podsetnik

Pokazivači i nizovi

U C programskom jeziku, pokazivač je promenljiva koja sadrži adresu neke memorijske lokacije (najčešće je to adresa neke promenljive ili niza nekog tipa). Pri definiciji pokazivača neophodno je navesti tip promenljive na koju pokazuje, odnosno čiju adresu sadrži. Najjednostavnija definicija pokazivača izlegleda na sledeći način:

```
type* pointer_name;
```

Memorijska adresa koju će da sadrži, odnosno memorijska adresa na koju će da pokazuje, se dodeljuje pokazivaču pomoću operatora &:

```
int a = 5;
int* pointer_to_a = &a;
```

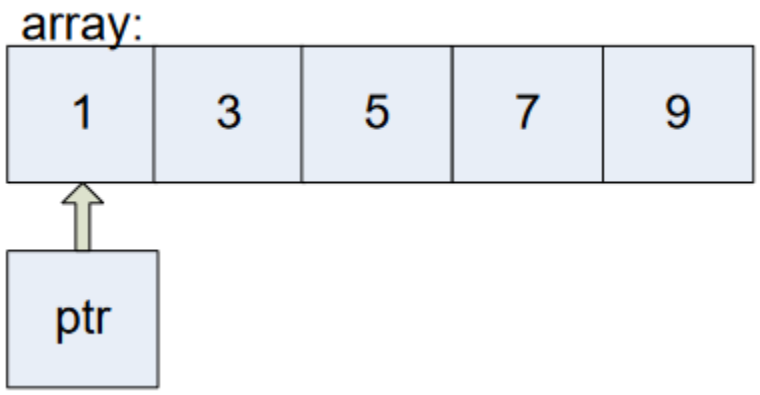
Niz se u programskom jeziku C može posmatrati kao blok memorije određene veličine za koji smatramo da se u njemu nalaze elementi istog tipa (iste veličine), jedan za drugim. Niz se, u najjednostavnijem obliku, definiše na sledeći način:

```
element_type array_name[dimension];
```

Nad nizovima možemo koristiti operator indeksiranja ([]) kako bismo preuzeli vrednost elementa na određenoj poziciji u nizu.

Niz, odnosno samo ime niza predstavlja njegovu adresu – možemo koristiti ime niza kao pokazivač na taj niz, ili da nekom pokazivaču dodelimo njegovu adresu, kako bi pokazivao na niz:

```
int array[] = { 1,3,5,7,9 };
int* ptr = array; /* <=> int* ptr = &array[0];
```



Pošto se ime niza ponaša kao pokazivač na taj niz, nekom *i*-tom elementu možemo pristupiti i pomoću operatora [], kao i računanjem adrese tog elementa u odnosu na početak niza pomoću aritmetike s pokazivačima, dereferenciranjem, tj. važi sledeće:

```
A[B] <=> *(A + B)
```

Primer

```
int array[] = { 1,3,5,7,9 };

int a = array[2];     /* pristup elementu na indeksu 2 i preuzimanje njegove
                         vrednosti korišćenjem operatora indeksiranja [] */

int b = *(array + 2); /* pristup elementu na indeksu 2 sračunavanjem njegove adrese
                         i preuzimanje vrednosti pomoću operatora * (operator
                         dereferenciranja) */

/* promenljive a i b sada imaju istu vrednost – 5 */
```

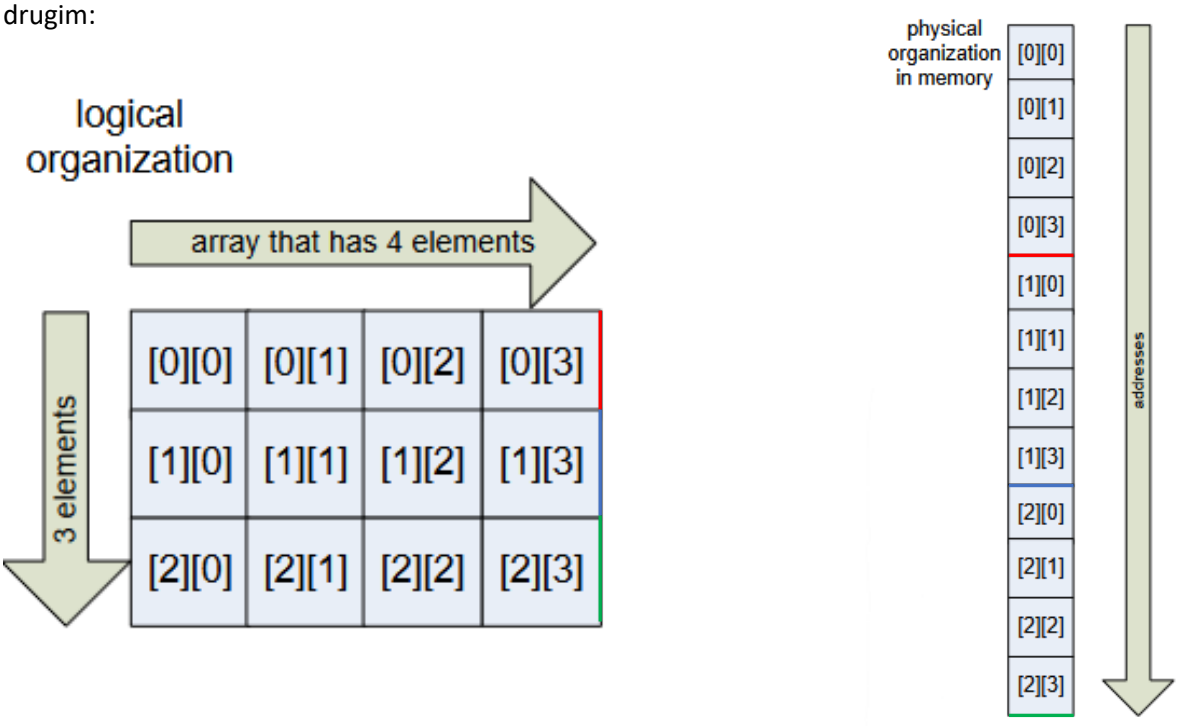
Višedimenzionalni nizovi u C progamsku jeziku su zapravo nizovi nizova. Primer definicije 2D niza:

```
#define N 3

#define M 4

int matrix[N][M]; /* matrix je niz od N elemenata, gde je svaki element zapravo niz od
                     M elemenata tipa int */
```

Logički posmatrano, ovi elementi su poređani u matricu. Fizički, u memoriji, oni su poređani jedan za drugim:



Elementu na poziciji (*i*, *j*) – *i*-ti red i *j*-ta kolona – se pristupa na sledeći način:

```
A[B][C] <=> *(*(A + B) + C)

<=>

D = *(A + B),

*(D + C)
```

Primer

```
int matrix[2][5] = { {1,3,5,7,9},
                     {0,2,4,6,8} };

int  a = matrix [1][2]; /* pristup elementu na poziciji (1,2) i preuzimanje njegove
                           vrednosti korišćenjem operatora indeksiranja [] */

int  b = *(*( matrix + 1) + 2); /* pristup elementu na poziciji (1,2) sračunavanjem
                                   njegove adrese i preuzimanje vrednosti pomoću
                                   operatora * (operator dereferenciranja)*/

/* Alternativni način: */

int* pr = *(matrix + 1); /* pokazivač na red 1 */

int  c = *(pr + 2);      /* element na indeksu 2, u redu 1 */

/* promenljive a, b i c sada imaju istu vrednost – 4 */
```

Primer u obradi audio signala – mikser 2x5

```
#define BLOCK_SIZE 256

#define N 2
#define M 5

float gain_matrix[N][M] =
{
    {0.5, 0.5, 0.25, 0.3, 0.3},
    {0.5, 0.5, 0.25, 0.3, 0.3}
};

float input_buffer[N][BLOCK_SIZE];
float output_buffer[M][BLOCK_SIZE];

void processing()
{
	int out; // iterate over output channels
	int in; // iterate over input channels
	int k; // iterate over samples

	for (out = 0; out < M; out++) // for every output channel
	{
		float* out_channel = *(output_buffer + out);
		for (in = 0; in < N; in++) // calculate sample values, coming from every
                                           input channel
		{
			float* in_channel = *(input_buffer + in);
			float* gain_row = *(gain_matrix + in);
			 float gain = *(gain_row + out);
			for (k = 0; k < BLOCK_SIZE; k++) // do for every sample
			{
				*(out_channel + k) = *(in_channel + k) * gain;
			}
		}
	}
}
```

Literatura:

1. Đukić, M., PPuRV predavanja 5 Pokazivači i nizovi, dostupno na mreži (https://www.rt-rk.uns.ac.rs/sites/default/files/materijali/predavanja/PPuRV_Predavanja_5_Pokazivaci_i_nizovi.pdf), [pristupljeno 24.10.2021.]