Vježba 5

Programiranje 1 (550) - ak. god. 2024./2025.

Nizovi

Niz (engl. *array*) je podatkovna struktura koji sadrži više podataka **istog tipa** i pohranjuje se na uzastopnim memorijskim lokacijama. Veličina niza se navodi prilikom deklaracije i nije je moguće promijeniti, što ih čini bržima od svih drugih podatkovnih struktura kojima se može mijenjati veličina.

> **Referencni tip**
>
> Niz je uvijek referencni tip bez obzira o kojem tipu podataka u nizu se radi. Varijabla niza na stogu ne sadrži same podatke, već referencu na objekt s podacima koji se nalazi na gomili.

Niz se deklarira na sljedeći način:

VB

```
Dim <imeVarijable> as <tip>()
```

Zagrade navedene iza tipa podatka govore da se radi o nizu, a ne o običnoj skalarnoj varijabli. Međutim, varijablu niza nije moguće koristiti odmah po deklaraciji, jer ju je, budući da se radi o referencnom tipu, potrebno instancirati. Npr. želimo li deklarirati i instancirati cijelih brojeva od 7 elemenata, to možemo napraviti na jedan od sljedećih načina

VB

```
Dim brojevi as Integer() = New Integer(6) {}

Dim brojevi As Integer() = New Integer(6) {0, 1, 2, 3, 4, 5, 6}

Dim brojevi As Integer() = {0, 1, 2, 3, 4, 5, 6}
```

Prva naredba deklarira i inicijalizira prazni niz sa 7 elemenata (pod prazni se misli da se za vrijednost elemenata postavlja pretpostavljena vrijednost za taj tip podataka; za numeričke tipove to je 0). U drugom primjeru se niz inicijalizira i odmah se postavljaju početne

vrijednosti njegovih elemenata, koje se nalaze u vitičastim zagradama. Zadnji primjer je skraćena verzija drugog zapisa i oni su ekvivalentni.

> **Indeksiranje**
>
> Pripazite na činjenicu da se **nizovi indeksiraju od nule**. Zato ovaj niz iz primjera, iako mu je navedena "veličina" 6, ima 7 elemenata.

Osim "običnih" nizova, mogu se pojaviti i nizovi s više dimenzija. Primjerice, dvodimenzionalne nizove možemo zamisliti kao matrice. Postupak je sličan, ali se veličina navodi u dvije dimenzije odvojene zarezom:

VB

```
Dim matrix As Integer(,) = New Integer(2, 2) {}
```

Pristupanje elementima niza

Elementima niza se pristupa preko njihovog indeksa (pozicija unutar niza). ***Indeksi započinju od 0***, tj. ako niz ima N elemenata, indeksi idu od 0 do $N-1$. Elementima dvodimenzionalnih nizova se pristupa s dva indeksa:

VB

```
brojevi(2)   ' pristupa trećem elementu niza (brojanje od 0)
matrix(1, 2) ' pristupa elementu u drugom retku, trećem stupcu
(brojanje od 0)
```

Svojstva nizova

Ukoliko je potrebno, rang niza (broj dimenzija) možemo dohvatiti korištenjem svojstva Rank na varijabli niza: `<varijablaNiza>.Rank`.

Broj elemenata niza se može dobiti korištenjem svojstva `Length`, koje vraća ukupni broj elemenata u nizu, bez obzira na broj dimenzija niza. Za jednodimenzionalne nizove, broj elemenata niza će biti jednak duljini niza: `<varijablaNiza>.Length`.

U slučaju višedimenzionalnih nizova, broj elemenata u pojedinoj dimenziji niza se može dobiti pozivanjem funkcije `GetLength` nad varijablom niza. Ta funkcija kao parametar prima redni broj dimenzije niza, koji počinje od 0. Dakle, za dvodimenzionalne nizove, parametar rednog broja dimenzije niza može imati vrijednost 0 ili 1:

VB

```
brojevi.GetLength(0) ' vraća 7
matrix.GetLength(1) ' vraća 3
```

Još jedna, posebno korisna funkcija kod rukovanja nizovima je `GetUpperBound`. Ona također kao parametar prima redni broj dimenzije niza, te vraća indeks zadnjeg elementa u nizu u toj dimenziji. Taj broj će uvijek biti za jedan manji nego onaj koji vrati funkcija `GetLength`. Funkcija `GetUpperBound` je posebno korisna u petljama za određivanje koliko puta se petlja treba izvršiti, primjerice ako je koristimo za prolazak kroz sve elemente nekog niza:

VB

```
brojevi.GetUpperBound(0) ' vraća 6
matrix.getUpperBound(1) ' vraća 2
```

Plitko i duboko kopiranje nizova

Kopiranje varijable niza (a analogno, i svih ostalih referencnih tipova podataka) ne kopira sam sadržaj objekta, već samo referencu:

VB

```
Dim niz as Integer() = {1, 2, 3, 4}
Dim kopija as Integer()

'Kopiranje varijable niza
kopija = niz

'Promjena podataka u kopiji
kopija(2) = 9
Console.WriteLine(niz(2)) 'Ispisuje 9
```

Gornji kod potvrđuje da kod ovakve vrste kopiranja, eventualna promjena kopiranog objekta mijenja i originalni objekt.

Ako želimo kopirati niz u novu varijablu koji bi podrazumijevao i kopiranje svih podataka u objektu, primjenjuje se tehnika koja se zove **plitko kopiranje** (engl. *shallow copy*). Za to se koristi ugrađena metoda `Clone()`:

VB

```
Dim niz as Integer() = {1, 2, 3, 4}
Dim kopijaNiza as Integer()

'Kopiranje niza
kopijaNiza = niz.Clone()

```

```
'Promjena podataka u kopiji
kopija(2) = 9

Console.WriteLine(kopijaNiza(2)) 'Ispisuje 9
Console.WriteLine(niz(2)) 'Ispisuje 3
```

Plitko kopiranje nizova

Plitkim kopiranjem niza čiji su elementi vrijednosnog tipa, stvaraju se dva objekta s elementima tog niza.
Plitkim kopiranjem niza čiji su elementi referencnog tipa, stvaraju se dvije varijable koje upućuju na isti objekt niza. Dakle, ne kopiraju se objekti na koje objekt koji se kopira upućuje.

Dubinskim kopiranjem (engl. *deep copy*) se kopira originalni objekt zajedno sa svim objektima na koje on upućuje.

Prosljeđivanje nizova po vrijednosti i po referenci

Nizovi se u proceduru mogu proslijediti po vrijednosti i po referenci, kao i sve ostale vrste varijabli. Ako niz prosljeđujemo po vrijednosti, to treba navesti u parametarskoj listi funkcije:

VB

```
Function SrednjaVrijednost(ByVal brojevi As Integer()) As Double
        ' Kod funkcije
End Function
```

Budući da je niz referencni tip podatka (dakle, objekt niza ne sadrži podatke već samo memorijsku lokaciju gdje se oni nalaze), prosljeđivanje niza u funkciju po vrijednosti (ključna riječ `VB ByVal` u parametarskoj listi funkcije) će samo kopirati varijablu niza, a ta kopija će opet pokazivati na isti set podataka u memoriji računala. Tako da bilo koja promjena nad originalnim podacima unutar funkcije će biti vidljiva i programskom kodu koji se nalazi izvan funkcije.

Ako se niz u funkciju proslijedi po referenci (`VB ByRef` modifikator umjesto `VB ByVal`) prosljeđuje funkciji samo memorijsku adresu gdje se podaci nalaze, čime pozvana funkcija dobiva potpunu kontrolu nad memorijskom adresom, pa je čak moguće i da se otpusti referenca, tj. postavi na vrijednost `VB Nothing`. Zato ovaj modifikator treba koristiti s oprezom.

For Each struktura ponavljanja

FOR EACH...NEXT struktura je konceptualno slična FOR...NEXT strukturi, a koristi se za jednostavan prolaz kroz sve elemente niza (ili neke druge zbirke). Primjerice, sljedeći kod:

VB

```vb
Dim brojevi As Integer() = {1, 2, 3, 4, 5}
For Each broj In brojevi
        Console.Write("{0} ", broj)
Next
Console.WriteLine()
```

je ekvivalentan kodu u kojem se postavlja brojač i gornja granica na zadnji element niza:

VB

```vb
Dim brojevi As Integer() = {1, 2, 3, 4, 5}
For brojac As Integer = 0 To brojevi.GetUpperBound(0)
        Console.Write("{0} ", brojevi(brojac))
Next
Console.WriteLine()
```

ali je prikladan jer ne moramo razmišljati o granicama niti o tipu podataka od kojih se sastoji niz.

Procedura s proizvoljnim brojem parametara

Moguće je definirati proceduru koja će primiti proizvoljan broj parametara korištenjem ključne riječi `ParamArray`. Da bi ga se koristilo, mora biti zadovoljeno sljedeće:

- može se koristiti samo jednom, kao zadnji parametar u parametarskoj listi
- mora biti proslijeđen po vrijednosti, a ne po referenci
- svi prethodni parametri moraju biti obavezni
  `ParamArray` će proizvoljan broj parametara interpretirati kao niz podataka **istog** tipa, pa zato kod njegove deklaracije ga treba navesti kao niz, kao u sljedećem primjeru koji prikazuje funkciju koja računa sumu proizvoljnog broja cijelih brojeva:

VB

```vb
Sub Main(args As String())
        Console.WriteLine(Suma(1, 2, 3, 4, 5)) ' Ispisuje 15
End Sub

Function Suma(ParamArray args() As Integer) As Integer
        Dim s As Integer = 0
        For Each arg In args
                s += arg
        Next
        Return s
End Function
```

Zadaci

1. Napišite program u kojem će korisnik unijeti prirodni broj N. Potom, kreirajte niz cijelih brojeva dužine N, u koji korisnik unosi brojeve. Kada je unos gotov, ispišite sve pozitivne brojeve u nizu koji su djeljivi s 3, uz mjesto (indeks) na kojem se nalaze.
2. Napišite proceduru Ispis() koja kao parametar prima niz brojeva, te ga ispisuje u konzoli, u jednoj liniji, odvojene znakom Tab (u VB.NET, znak Tab se dobiva korištenjem konstante vbTab). Zatim tu proceduru preopteretite tako da kao parametar može biti i dvodimenzionalni niz brojeva, te implementirajte da ga ispisuje kao matricu (brojevi odvojeni znakom Tab).
   Navedene procedure možete koristiti u sljedećim zadacima za ispis sadržaja nizova.
3. Napišite funkciju koja kao parametar prima niz cijelih brojeva te vraća njihovu sumu, te drugu funkciju koja vraća prosječnu vrijednost elemenata niza. Rad funkcije testirajte tako da u proceduri Main() korisnik unese broj elemenata niza, te nakon toga svaki od članova niza. Ispišite sumu i prosjek korištenjem napisanih funkcija.
4. Napišite funkciju koja će generirati niz od N slučajno odabranih brojeva u rasponu od -5 do 5 (N unosi korisnik). Potom napravite novi niz u kojem ćete negativne brojeve iz polaznog niza kvadrirati, a pozitivne korjenovati. Na kraju ispišite početni (generirani) niz, te dobiveni niz.
5. Napišite funkciju koja zbraja dvije matrice cijelih brojeva. U proceduri Main() zatražite od korisnika unos dvije matrice dimenzija 2x2, te na kraju ispišite njihov zbroj.
6. Napišite funkciju koja prima kao parametar dva niza – niz ocjena, te niz vrijednosti ECTS bodova za predmete. Na kraju funkcija računa i vraća otežani (ponderirani) prosjek (w_i je težina – broj ECTS bodova, a k_i je ocjena). U glavnom programu korisnik unosi broj ocjena koje će unijeti, te nakon toga unosi pojedinačne ocjene i ECTS bodove. Pozovite funkciju s unesenim vrijednostima te ispišite rezultat.

$$prosjek = \frac{\sum_{i=1}^{N} w_i k_i}{\sum_{i=1}^{N} w_i}$$

7. Napišite proceduru koja će primati **proizvoljan broj** parametara tipa integer, te na kraju ispisati njihov prosjek. Zatim proceduru prekrcajte tako da osim integera na isti način može primati i realne brojeve. Testirajte proceduru s različitim brojem ulaza.
8. Napišite funkciju koja prima niz kao argument i vraća transponiranu matricu ako je niz dvodimenzionalan. Prekrcajte funkciju tako da ako primi niz koji je jednodimenzionalan da ga vraća isti niz obrnutim redoslijedom. Testirajte funkciju s obje vrste unosa.
9. Napišite funkciju koja prima dva parametra: prvi je niz realnih brojeva, a drugi je prirodni broj. Drugi parametar neka bude opcionalan s predefiniranom vrijednošću od e (baza prirodnog logaritma). Funkcija treba vratiti novi niz brojeva od kojih je svaki broj logaritam broja iz ulaznog niza na odgovarajućem mjestu, po zadanoj bazi. (Logaritam se računa funkcijom `Math.Log(broj, baza)`. Konstanta e je definirana kao `Math.E`.