

Verem (LIFO) adattípus:

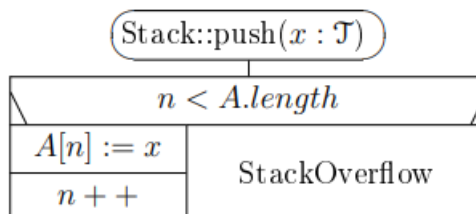
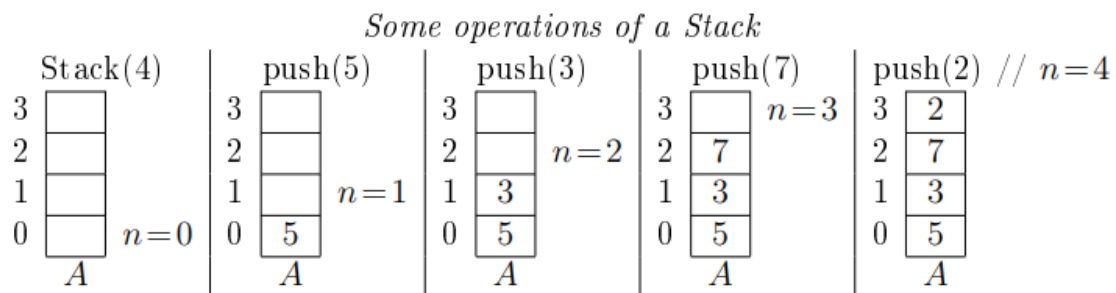
A vermet az $A[0..m]$ tömbben ábrázoljuk (m -et a létrehozáskor meg kell adni), a verem elemei az $A[0..n]$ résztömbben helyezkednek el, ha $n > 0$.

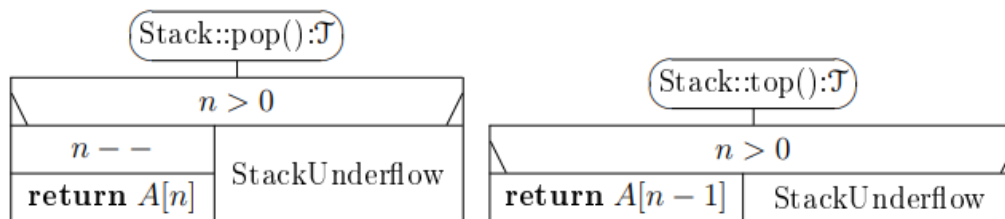
Megjegyzés: kicsit kényelmetlen, hogy a verem konstruktorában kötelező megadni a verem méretét. A tavalyi magyar nyelvű jegyzetben az automatikusan „nyújtózkodó” verem van definiálva, az kényelmesebb, mert nem kell előre tudni, mekkora veremre lesz szükségünk. Viszont annak tömbje 1-től van indexelve, ezért most itt inkább az angol jegyzetben szereplő vermet használom. Ha Tibi felteszi az aktuális jegyzetet, az órán érdemes az abban szereplőt bemutatni.

UML ábra az angol jegyzetből:

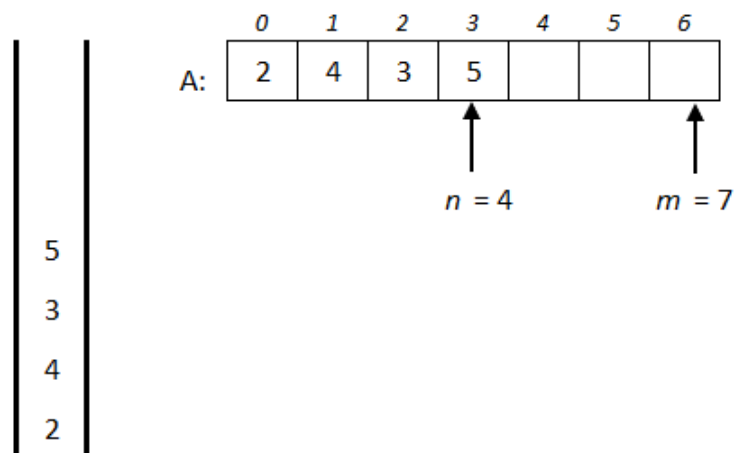
Stack	
$-A : \mathcal{T}[]$	// \mathcal{T} is some known type ; $A.length$ is the max. size of the stack
$-n : \mathbb{N}$	// $n \in 0..A.length$ is the actual size of the stack
$+ \text{Stack}(m : \mathbb{N}) \{ A := \text{new } \mathcal{T}[m] ; n := 0 \}$ // create an empty stack $+ \text{push}(x : \mathcal{T})$ // push x onto the top of the stack $+ \text{pop}() : \mathcal{T}$ // remove and return the top element of the stack $+ \text{top}() : \mathcal{T}$ // return the top element of the stack $+ \text{isFull}() : \mathbb{B} \{ \text{return } n = A.length \}$ $+ \text{isEmpty}() : \mathbb{B} \{ \text{return } n = 0 \}$ $+ \text{setEmpty}() \{ n := 0 \}$ // reinitialize the stack $+ \sim \text{Stack}() \{ \text{delete } A \}$	

Műveletek szemléltetése a jegyzetből:





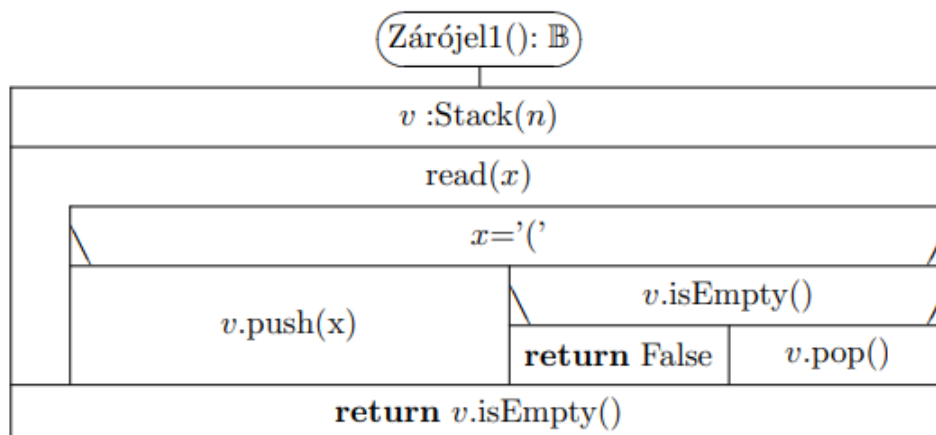
Rajzos szemléltetés:



Veremmel megoldható feladatok:

- Adott egy zárójelekből álló, legfeljebb n hosszú karaktersorozat a bemeneten. Olvassuk be, és döntsük el róla, hogy helyes zárójelezést határoz-e meg! (Vagyis, hogy párba állíthatók-e a zárójelek úgy, hogy minden nyitó zárójelnek van egy olyan csukó zárójel a párja, amely később következik a sorozatban.)

Megoldás verem segítségével:



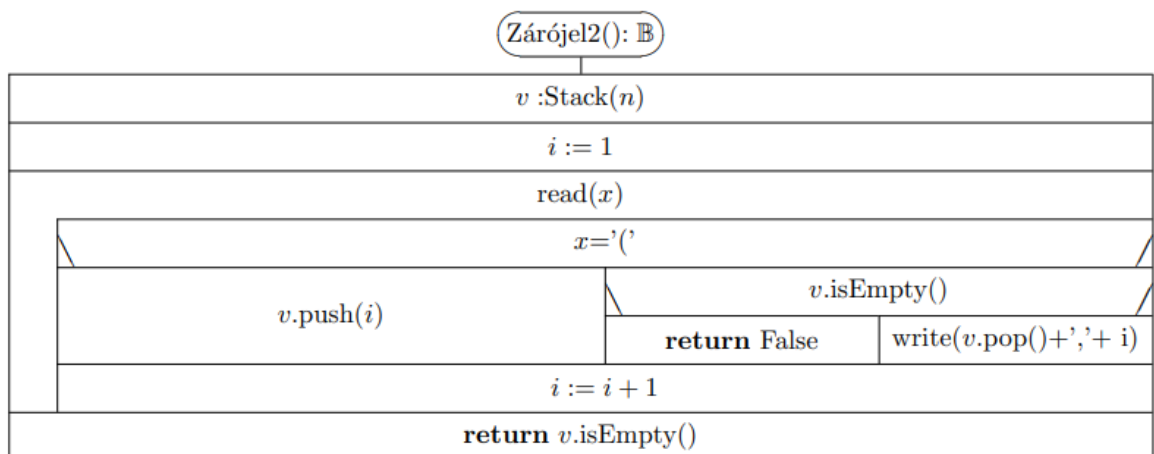
Fejléc helyesebben: Zárójel1($n:\mathbb{N}$): \mathbb{B}

Megjegyzések:

- A beolvasáshoz az előadásjegyzetben is használt `read(&x : T):B` függvényt használtuk, de ettől eltérhetünk. Ez a függvény beolvassa `x`-be a következő értéket a bemenetről, és ha ez sikeres, akkor igaz, ha pedig végére értünk az inputnak, akkor hamis lesz.
- Vegyük észre, hogy a fenti módszerhez valójában nincs is szükség verem használatára. Elég lenne egyetlen számláló változóval nyilvántartani azt, hogy eddig mennyivel több nyitó zárójellel találkoztunk, mint csukó zárójellel.
- Módosítsuk úgy a feladatot úgy, hogy tényleg szükség legyen a veremre: Ezúttal írassuk ki az összetartozó zárójelpárok indexeit!

2. Adott egy zárójelekből álló, legfeljebb n hosszú karaktersorozat a bemeneten. Olvassuk be, és döntsük el róla, hogy helyes zárójelezést határoz-e meg! Írassuk ki az összetartozó zárójelpárok indexeit!

Megoldás:



Fejléc helyesebben: `Zárójel2(n:N):B`