Datum odevzdání: 24.01.2022

Datové typy

Kuba kacálek ,ondřej Skutil

2022

# Datové typy

## Jednosměrný seznam

Má vždy dva prvky hodnotu a hlavu(hodnotu) odkazující na další v seznamu

Nemusí se celý přepisovat při odebrání. Při odebrání bude prvek k-1 na k+1.

## Zásobník

Pracuje jen s prvky zeshora

Pokud chceme sehnat prvek s nějakým indexem musíme si odebírané hodnoty dávat jinám, aby jsme je pak mohli vrátit zpět

## Fronta(FIFO)

Pridava prvky zespoda a bere zeshora

# Lineární vs Binary Search

## Lineární

* dá používat na cokoli co se dá procházet
* musí projít celý řetězec, a proto je pomalejší

## Binary

* Řetězec musí být indexovaný pracuje se s čísly
* Používá se, když je řetězec seřazený
* Je rychlejší, protože vždy část řetězce půlí, a když zjistí, že je střed menší, či větší, tak si veme část, v které je hledaný prvek obsáhlý a dále půlí vybranou část než splňuje podmínku, že hledaný prvek je středem půlení.

## Ukázkový kód(C#)

### Binary

static int binarySearch(int[] arr, int l, int r, int h)

{

// Pokud zbývá jeden prvek končíme

if (r >= l)

{

// Zjistíme střed vybrané části

int mid = l + (r - l) / 2;

// Pokud byl prvek nalezen, končíme

if (arr[mid] == h)

return mid;

// Pokud je středový prvek větší než hledaný. Prvek musí být v na levé straně pole

if (arr[mid] > h)

return binarySearch(arr, l, mid - 1, h);

// Pokud je středový prvek menší než hledaný. Prvek musí být v na pravé straně pole

return binarySearch(arr, mid + 1, r, h);

}

// Prvek nebyl nalezen

return -1;

### }

### Lineární

static int Linear<T>(IEnumerable<T> arr,T h)

{

int index = 0;

// Procházím pole hledám prvek

foreach(var t in arr)

{

//Nasel jsem prvek vrátím index

if (h.Equals(t)) { return index; }

//Zvětšuji index s každým průchodem

index++;

}

// Prvek nebyl nalezen

return -1;

### }

# Odkaz na Github

<https://github.com/Kaciis/abstraktniTypy>