

Kód ITMS projektu: 26220220123



**Európska únia**  
Európsky fond regionálneho rozvoja

## Výstup: V2.2.8 ImplKomIT

**Ad-hoc integračný komponent komponent predmetov z oblasti vied informačných a komunikačných technológií.**

**VizAlgo - nástroj pre vizualizáciu algoritmov a údajových štruktúr**

<b>Riešiteľský kolektív</b>	Slavomír Šimoňák
-----------------------------	------------------

### História dokumentu:

Verzia	Autor(i)	Dátum	Sumár zmien
1	Slavomír Šimoňák	20.09.14	Príprava príručky k nástroju VizAlgo.

## Obsah

1.Funkcia nástroja.....	3
2.Súpis obsahu dodávky.....	3
3.Inštalácia programu.....	3
4.Použitie programu.....	3
5.Dodané vizualizácie.....	3
Modul binarySearchTree.....	5
Modul bubblesort.....	6
Modul chainmatrix.....	7
Modul heapsort.....	8
Modul insertsort.....	9
Modul mergesort.....	10
Modul MinMax.....	11
Modul queue.....	12
Modul quicksort.....	13
Modul radixsort.....	14
Modul selectsort.....	15
Modul shellsort.....	16
Modul stack.....	17
Modul treeTraversal.....	18

## 1. Funkcia nástroja

VizAlgo je podporný nástroj pre skvalitnenie výučby algoritmov a údajových štruktúr formou vizualizácie. Medzi hlavné prednosti využitia vizualizácií patrí vyjadrenie konceptov vizuálnou formou, podpora praktických foriem vyučovania, zvýšenie pozornosti a zlepšenie komunikácie medzi vyučujúcim a študentmi. Interaktívne vizualizácie podporujú experimentovanie a objavovanie ideí s ohľadom na individuálne potreby študentov.

## 2. Súpis obsahu dodávky

Nástroj je dodávaný vo forme archívu *VizAlgo.zip* a súboru *VizAlgo\_navod.pdf*, ktorý predstavuje tento stručný návod. Archív obsahuje tieto časti:

- súbor *VizAlgo.jar* – samotná aplikácia,
- katalóg *plugins* – obsahujúci zásuvné moduly dodaných vizualizácií,

## 3. Inštalácia programu

Program nie je potrebné inštalovať, stačí rozbaľiť dodaný archív. Program vyžaduje pre svoju činnosť behové prostredie Java (JRE). Odporúčaná je verzia 1.7 alebo novšia prostredia JRE. Spustenie aplikácie je následne realizované pomocou súboru *VizAlgo.jar*.

## 4. Použitie programu

Použitie programu je vďaka prehľadnému používateľskému rozhraniu (GUI) intuitívne. Prostredie programu je znázornené na obrázku Obr. 1 a pozostáva z niekoľkých panelov:

- panel vstupov,
- pseudokód algoritmu,
- vizualizácia algoritmu,
- informácie,
- ovládací panel.

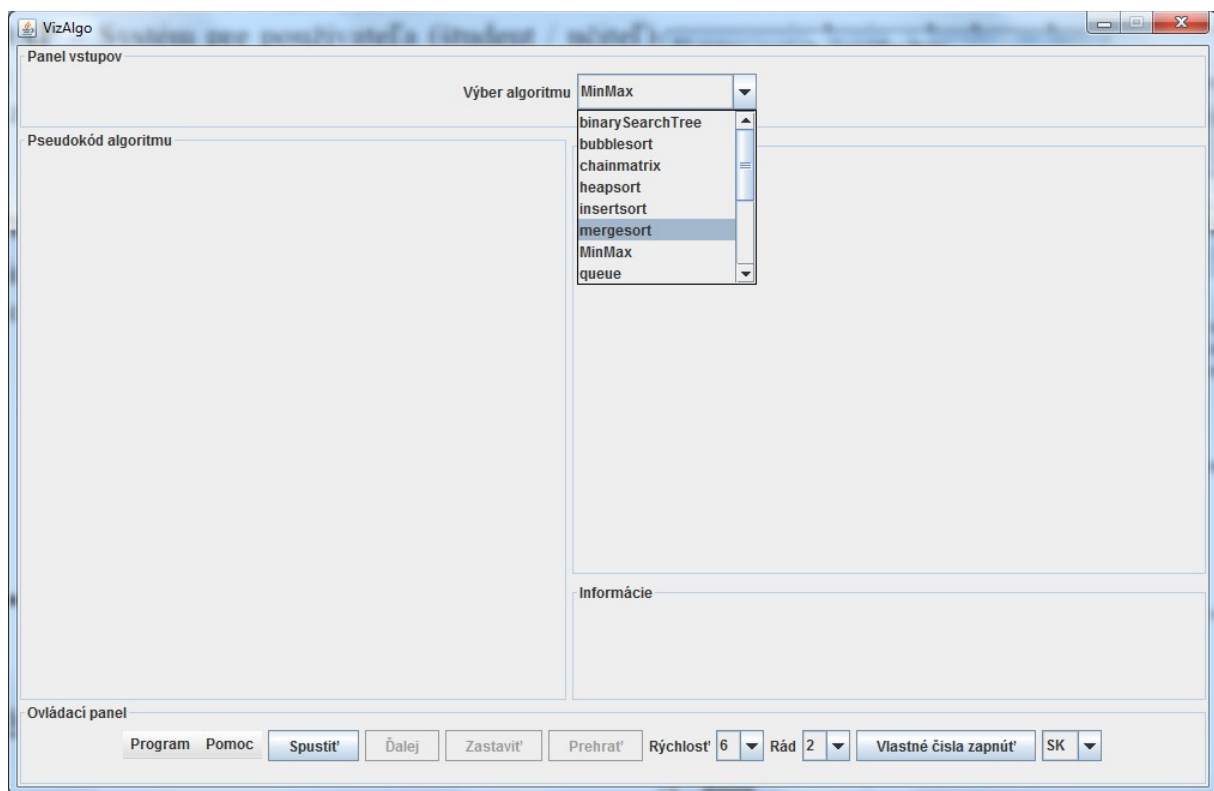
Výber vizualizácie sa realizuje v rámci panela vstupov prostredníctvom rozbaľovacieho zoznamu. Pre ovládanie vizualizácie zvoleného algoritmu sa používajú prvky ovládacieho panela v spodnej časti okna aplikácie.

## 5. Dodané vizualizácie

K súčasnej verzii nástroja VizAlgo sú dodané nasledujúce zásuvné moduly:

- `binarySearchTree`
- `bubblesort`
- `chainmatrix`
- `heapsort`
- `insertsort`

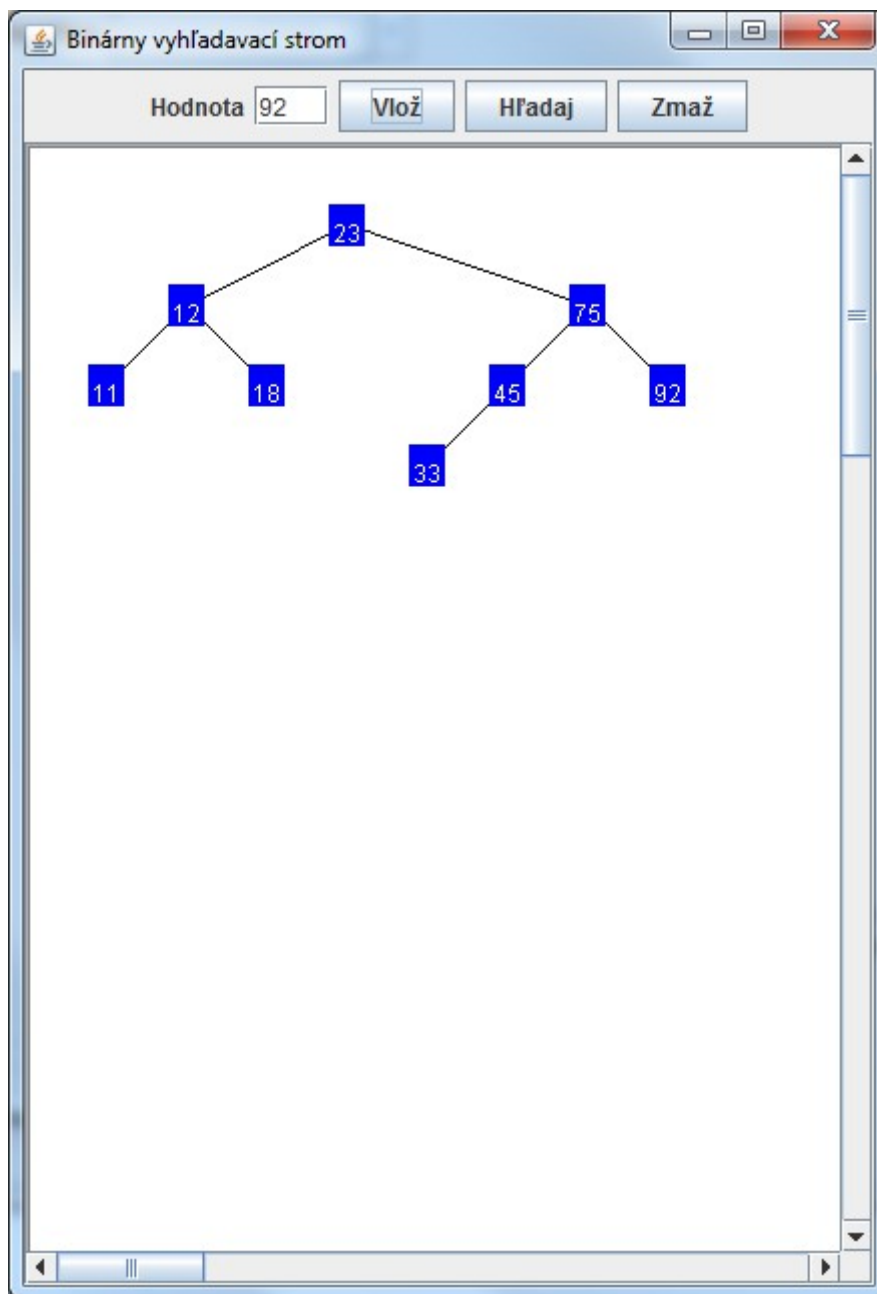
- mergesort
- MinMax
- queue
- quicksort (dve verzie)
- radixsort
- selectsort
- shellsort
- stack
- treeTraversal



Obr. 1: Prostredie nástroja VizAlgo

## Modul binarySearchTree

Modul obsahuje vizualizáciu binárneho vyhľadávacieho stromu (BVS), Obr. 2. Vizualizácia sa po výbere spustí v samostatnom okne.



Obr. 2: Binárny vyhľadavací strom (BVS)

## Modul bubblesort

Modul obsahuje vizualizáciu jednoduchého algoritmu bublinkového triedenia (Bubblesort), Obr. 3.

The screenshot shows the VizAlgo application interface for the Bubblesort algorithm. The window title is "VizAlgo".

**Panel vstupov:** "Výber algoritmu" is set to "bubblesort" and "Zobrazenie" is set to "Klasické".

**Pseudokód algoritmu:**

```
for (pass = 1; pass < n; pass++) {  
  for (i = 0; i < n-pass; i++) {  
    if (x[i] > x[i+1]) {  
      temp = x[i];  
      x[i] = x[i+1];  
      x[i+1] = temp;  
    }  
  }  
}
```

**Vizualizácia algoritmu:** A horizontal array of numbers is shown: 3, 43, 55, 81, 54, 18, 6, 84, 65, 36. The number 81 is highlighted in green, and 54 is highlighted in red. A red arrow labeled 'i' points to the position of 81. Below the array, it says "Hodnoty premenných: pass: 1 i: 3".

**Informácie:** Metóda: Bublínkové triedenie (ang. bubble sort) je implementačne jednoduchý algoritmus pracujúci na báze porovnania a výmeny prvkov. V rámci triedenej postupnosti prvkov, sú porovnávané vždy dva susedné prvky. Ak prvky nie sú v správnom poradi, sú vzájomne zamenené. Zložitosť:  $T(n) = O(n^2)$ . V prípade usporiadaného poľa na vstupe  $T(n) = O(n)$ .

**Ovládací panel:** Buttons for "Program", "Pomoc", "Spustiť", "Ďalej", "Zastaviť", "Prehrať", "Rýchlosť" (set to 6), "Rád" (set to 2), "Vlastné čísla zapnúť", and "SK".

Obr. 3: Vizualizácia algoritmu Bubblesort

## Modul chainmatrix

Modul obsahuje vizualizáciu algoritmu pre nájdenie minimálneho počtu operácií násobenia postupnosti matíc (chainmatrix) s využitím metódy dynamického programovania. Najprv je programom vyžiadany počet matíc, potom ich rozmery. Činnosť je zachytená na Obr. 4.

The screenshot shows the VizAlgo application window. The interface is divided into several sections:

- Panel vstupov:** A dropdown menu for 'Výber algoritmu' is set to 'chainmatrix'. To its right, 'Rozmery matíc:' are input as 10, 20, 50, 1, and 100.
- Pseudokód algoritmu:** A block of pseudocode is shown. The line `if(q < m[i][j])` is highlighted in yellow.
- Vizualizácia algoritmu:** This section displays the current state of the algorithm:
  - Počet matíc: n = 4
  - Rozmery matíc: A table with columns r[0], r[1], r[2], r[3], r[4] and values 10, 20, 50, 1, 100. The values 10, 1, and 100 are highlighted in red.
  - Hodnoty premenných: l: 3 i: 1 j: 4 k: 3 q: 2200
  - Table of m values:

m[1][1]=0	m[2][2]=0	m[3][3]=0	m[4][4]=0
m[1][2]=10000	m[2][3]=1000	m[3][4]=5000	
m[1][3]=1200	m[2][4]=3000		
m[1][4]=2200			
  - min( 23000 , 65000 , 2200 )
- Informácie:** A text box explaining the dynamic programming technique and the complexity  $T(n) = O(n^3)$ .
- Ovládací panel:** A control bar at the bottom with buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', 'Rýchlosť' (set to 6), 'Rád' (set to 2), 'Vlastné čísla zapnúť', and 'SK'.

Obr. 4: Určenie ceny pre násobenie postupnosti matíc

## Modul heapsort

Modul obsahuje vizualizáciu algoritmu triedenia haldou (heap). Činnosť algoritmu, vrátane zobrazenia poľa triedených prvkov vo forme špeciálneho binárneho stromu s vlastnosťou haldy je zachytená na Obr. 5.

The screenshot shows the VizAlgo application window. At the top, the algorithm is set to 'heapsort'. The 'Pseudokód algoritmu' section contains the following code:

```
BuildHeap(Array);
for(i = n; i > 1; i--)
{
    Swap(Array, 1, i);
    Heapify(Array, 1, i-1);
}

Heapify(Array, int i, int j)
{
    if(2*i > j) return;
    if(2*i == j) SwpIdx = 2*i;
    else if(Array[2*i] > Array[2*i+1]) SwpIdx = 2*i;
    else SwpIdx = 2*i+1;
    if(Array[i] < Array[SwpIdx])
    {
        Swap(Array, i, SwpIdx);
        Heapify(Array, SwpIdx, j);
    }
}
```

The 'Vizualizácia algoritmu' section displays an array of numbers: 23, 08, 34, 36, 50, 63, 67, 73, 78, 88. The values 23 and 34 are highlighted in green. Below the array, a binary tree is shown with the same values. The root node is 23, its left child is 08, and its right child is 34. Node 08 has children 36 and 50. Node 34 has children 63 and 67. Node 36 has children 73 and 78. Node 50 has child 88. The values 36, 50, 63, 67, 73, 78, and 88 are shown in pink.

The 'Informácie' section contains the following text:

následne zmenšená o tento jeden prvok a je na nej znova vytvorená halda volaním Heapify(Array,1,i-1). Takto sa na uvoľnených pozíciách v smere od konca poľa postupne ukladajú odobraté prvky počnúc najväčším a formuje sa utriedená postupnosť prvkov.  
Zložitosť:  $T(n) = O(n \cdot \log n)$  v najhoršom prípade.

The 'Ovládaci panel' at the bottom includes buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', 'Rýchlosť' (set to 6), 'Rád' (set to 2), 'Vlastné čísla zapnúť', and 'SK'.

Obr. 5: Triedenie haldou



## Modul insertsort

Modul obsahuje vizualizáciu algoritmu triedenia priamym vkladáním (insertsort). Činnosť algoritmu je zachytená na Obr. 6.

The screenshot shows the VizAlgo application interface. At the top, there is a 'Výber algoritmu' dropdown menu set to 'insertsort'. Below this, the 'Pseudokód algoritmu' section displays the following code:

```
for (i = 1; i < length[A]; i++) {  
    value = A[i]  
    j = i - 1  
    while j >= 0 && A[j] > value {  
        A[j+1] = A[j];  
        j = j - 1;  
    }  
    A[j+1] = value;  
}
```

The line `j = j - 1;` is highlighted in yellow. To the right, the 'Vizualizácia algoritmu' section shows a sequence of numbers: 46, 87, 98, 72, 42, 85, 66, 62, 90. Above the numbers 87 and 98 are labels 'j' and 'i' respectively, with arrows pointing down to the corresponding numbers. The number 87 is in a green box, and 98 is in a red box. Below the numbers, it says 'Hodnoty premenných: value: 54 i: 3 j: 1'. At the bottom, the 'Ovládací panel' contains buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', 'Rýchlosť' (set to 6), 'Rád' (set to 2), 'Vlastné čísla zapnúť', and 'SK'.

**Informácie**  
Metóda: Triedenie vkladáním (insert sort) je jednoduchý algoritmus využívajúci operáciu porovnania prvkov triedenej postupnosti. Vkladá nový prvok do utriedeného poľa na správnu pozíciu. Opakuje tento proces, pokiaľ nie sú takto vložené všetky prvky.  
Zložitosť:  $T(n) = O(n^2)$ . V prípade usporiadaného poľa na vstupe  $T(n) = O(n)$ .

Obr. 6: Triedenie vkladáním

## Modul mergesort

Modul obsahuje vizualizáciu algoritmu triedenia zlučováním (mergesort), využívajúceho dekompozíciu (metóda rozdeľuj a panuj). Činnosť algoritmu je zachytená na Obr. 7.

The screenshot shows the VizAlgo application window. At the top, the algorithm 'mergesort' is selected. The left pane displays the pseudocode for MergeSort, with the 'if(n > 1)' line highlighted in yellow. The right pane shows a visualization of the array [23, 46, 47, 70, 90, 59, 1, 79, 81, 37] being processed. The array is shown in three rows, with elements being compared and swapped. The bottom control panel includes buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', a speed slider set to 6, a row selector set to 2, and a checkbox for 'Vlastné čísla zapnúť'.

```
MergeSort(A[])
{
  if(n > 1)
  {
    MergeSort[A[1 ... n/2]];
    MergeSort[A[n/2 + 1 ... n]];
    Merge(A[1 ... n]);
  }
}
```

23	46	47	70	90	59	1	79	81	37	
23	46	47	90	70		59	1	79	81	37
23	46	47	90	70		59	1			
			90	70						

**Informácie**  
MergeSort je stabilný triediaci algoritmus typu rozdeľ a panuj s asymptotickou zložitosťou  $O(n \log n)$ . Pracuje na báze zlievania už roztriedených častí poľa za pomoci dodatočného poľa veľkosti  $n$ . MergeSort bol vynájdený v roku 1945 Johnom von Neumannom.

Obr. 7: Triedenie zlučováním

## Modul MinMax

Modul obsahuje vizualizáciu algoritmu pre nájdenie minimálneho a maximálneho prvku postupnosti s využitím dekompozície. Jedná sa o rekurzívny algoritmus a pre zjednodušenie pochopenia jeho činnosti je zobrazovaný aj zásobník volaní. Činnosť algoritmu je zachytená na Obr. 8.

The screenshot shows the VizAlgo application interface. The 'Výber algoritmu' dropdown is set to 'MinMax'. The 'Pseudokód algoritmu' section displays the following code:

```
pair MinMax(A[], int L, int H) {
    pair ret1, ret2, ret;
    if(L == H)
    {
        ret.min = A[L];
        ret.max = A[L];
    }
    else if(L == H-1)
    {
        ret.min = MIN(A[L], A[H]);
        ret.max = MAX(A[L], A[H]);
    }
    else
    {
        ret1 = MinMax(A, L, (L+H)/2);
        ret2 = MinMax(A, (L+H)/2+1, H);
        ret.max = MAX(ret1.max, ret2.max);
        ret.min = MIN(ret1.min, ret2.min);
    }
    return ret;
}
```

The 'Vizualizácia algoritmu' section shows an array of values: 21, 61, 5, 96, 86, 18, 83, 42, 20, 10. The values 18 and 10 are highlighted in green. Below the array, the following recursive calls are listed:

- MinMax L: 0 H: 9
- MinMax L: 0 H: 4 Min: 5 Max: 96
- MinMax L: 0 H: 2 Min: 5 Max: 61
- MinMax L: 0 H: 1 Min: 21 Max: 61
- MinMax L: 2 H: 2 Min: 5 Max: 5
- MinMax L: 3 H: 4 Min: 86 Max: 96
- MinMax L: 5 H: 9

The 'Zásobník volaní' section shows: MinMax(A,0,9) MinMax(A,5,9)

The 'Informácie' section contains the following text:

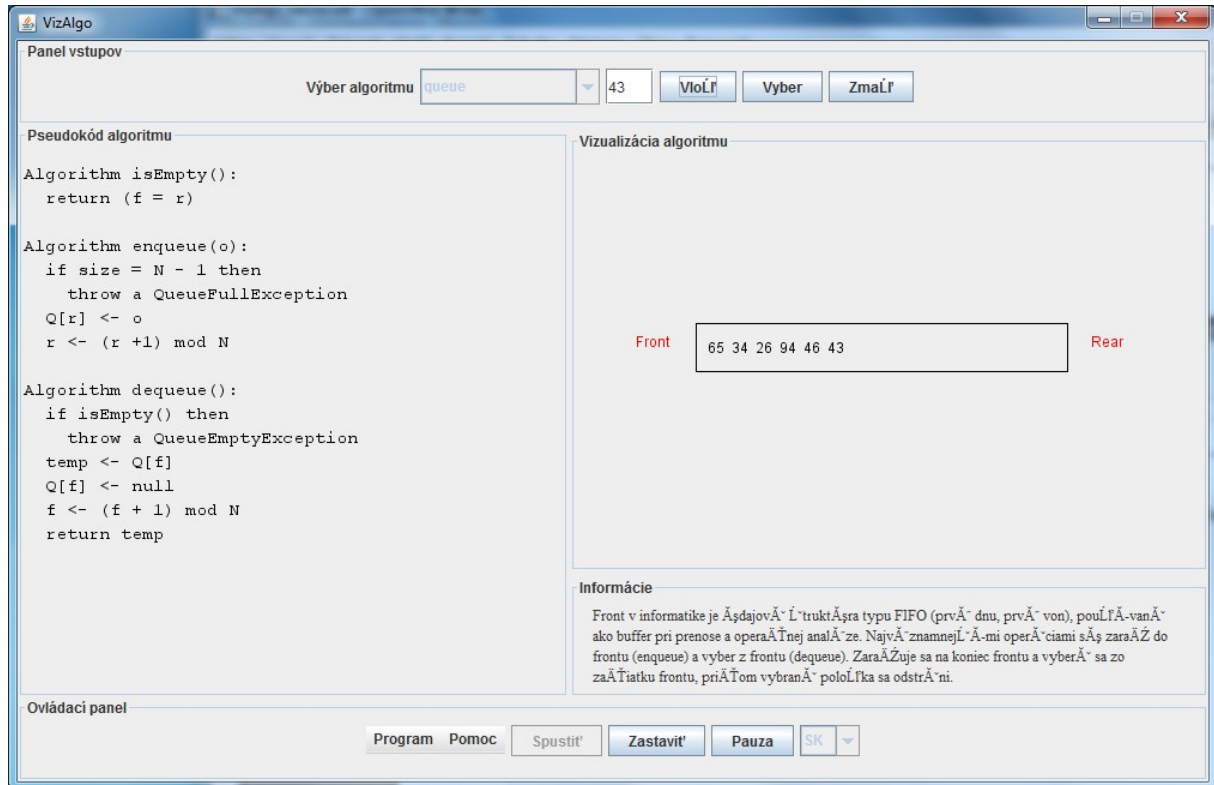
Metóda: rekurzívna aplikácia procedúry MinMax s využitím techniky Rozdeľuj a panuj.  
Procedúrou je pri každom volaní vrátená dvojica prvkov (minimálny a maximálny) z príslušného intervalu.  
Zložitosť:  $T(n) = 3(n/2) - 2$ .

The 'Ovládací panel' at the bottom includes buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', 'Rýchlosť' (set to 6), 'Rád' (set to 2), 'Vlastné čísla zapnúť', and 'SK'.

Obr. 8: Algoritmus MinMax

## Modul queue

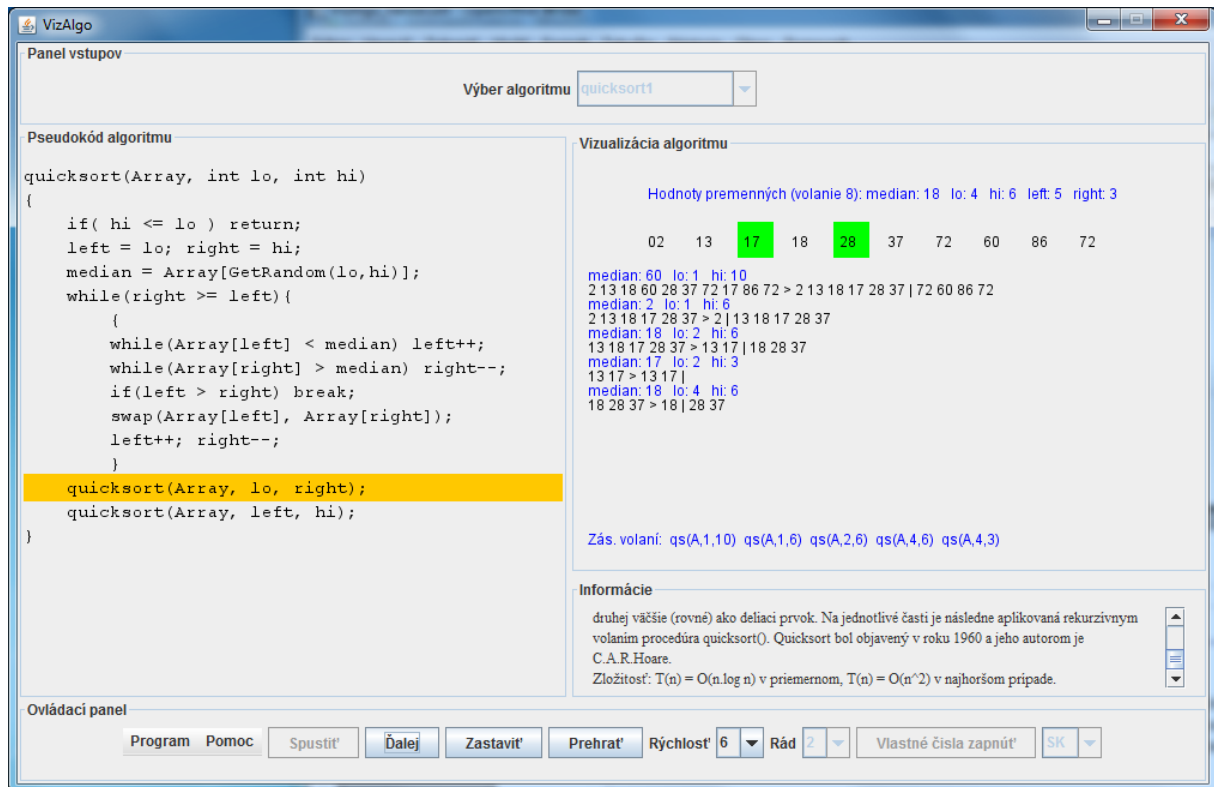
Modul obsahuje vizualizáciu činnosti údajovej štruktúry front (queue) typu FIFO (First In First Out). Vizualizácia je zachytená na Obr. 9.



Obr. 9: Údajová štruktúra front

## Modul quicksort

Modul obsahuje vizualizáciu činnosti efektívneho algoritmu triedenia s využitím dekompozície známeho pod názvom rýchle triedenie (quicksort). Vizualizácia činnosti algoritmu vrátane zásobníka volaní a jednotlivých delení postupnosti na základe zvoleného prvku (median) je zachytená na Obr. 10.



The screenshot shows the VizAlgo application interface for the quicksort algorithm. It is divided into several sections:

- Panel vstupov:** A dropdown menu labeled "Výber algoritmu" is set to "quicksort1".
- Pseudokód algoritmu:** A code editor showing the implementation of the quicksort function. The line `quicksort(Array, lo, right);` is highlighted in yellow.
- Vizualizácia algoritmu:** A visualization area showing the current state of the array and variables. The array is `02 13 17 18 28 37 72 60 86 72`, with `17` and `28` highlighted in green. Below the array, the current state of variables is shown: `median: 60 lo: 1 hi: 10`, `2 13 18 60 28 37 72 17 86 72 > 2 13 18 17 28 37 | 72 60 86 72`, `median: 2 lo: 1 hi: 6`, `2 13 18 17 28 37 > 2 | 13 18 17 28 37`, `median: 18 lo: 2 hi: 6`, `13 18 17 28 37 > 13 17 | 18 28 37`, `median: 17 lo: 2 hi: 3`, `13 17 > 13 17 |`, `median: 18 lo: 4 hi: 6`, and `18 28 37 > 18 | 28 37`. Below this, the call stack is shown: `Zás. volaní: qs(A,1,10) qs(A,1,6) qs(A,2,6) qs(A,4,6) qs(A,4,3)`.
- Informácie:** A text box providing information about the algorithm: "druhej väčšie (rovné) ako deliaci prvok. Na jednotlivé časti je následne aplikovaná rekurzívny volaním procedúra quicksort(). Quicksort bol objavený v roku 1960 a jeho autorom je C.A.R.Hoare. Zložitosť:  $T(n) = O(n \log n)$  v priemernom,  $T(n) = O(n^2)$  v najhoršom prípade."
- Ovládací panel:** A control panel at the bottom with buttons for "Program", "Pomoc", "Spustiť", "Ďalej", "Zastaviť", "Prehrať", "Rýchlosť" (set to 6), "Rád" (set to 2), "Vlastné čísla zapnúť", and "SK".

Obr. 10: Rýchle triedenie (quicksort)

## Modul radixsort

Modul obsahuje vizualizáciu činnosti algoritmu triedenia bez využitia operácie vzájomného porovnania triedených prvkov (radixsort). Porovnávané sú len jednotlivé časti (číslce, symboly) na zodpovedajúcich pozíciách a podľa výsledku týchto porovnaní sú triedené prvky umiestňované do príslušných frontov ( $Q[i]$ ). Vizualizácia činnosti algoritmu je zachytená na Obr. 11.

The screenshot shows the 'VizAlgo' application window. At the top, there is a 'Panel vstupov' (Input Panel) with a dropdown menu 'Výber algoritmu' (Select algorithm) set to 'radixsort'. Below this is the 'Pseudokód algoritmu' (Algorithm pseudocode) section, which contains the following code:

```
begin
  for j <- k step-1 until 1 do
    begin
      while QUEUE != EMPTY do
        begin
          move Ai from QUEUE into Q[aij];
        end
      end
      for l <- 0 until m-1 do
        begin
          concat Q[l] to the end of QUEUE;
          make Q[l] empty;
        end
      end
    end
  end
end
```

The line 'move A<sub>i</sub> from QUEUE into Q[a<sub>ij</sub>];' is highlighted in yellow. To the right of the pseudocode is the 'Vizualizácia algoritmu' (Algorithm visualization) section. It shows a 'QUEUE' with the values 22, 42, 72, 93, 93, 64, 14, 15, 07, 78. The value '07' is highlighted in green. Below the queue, it says '2.krok -- pozícia: 1' and lists the contents of buckets Q[0] through Q[9]:

- Q[0]: 07
- Q[1]: 14 15
- Q[2]: 22
- Q[3]: 42
- Q[4]: 42
- Q[5]: 64
- Q[6]: 64
- Q[7]: 72
- Q[8]: 93
- Q[9]: 93 93

At the bottom right of the visualization area is the 'Informácie' (Information) section, which contains the following text:

algoritmus zaraďuje prvky do skupín podľa posledného symbolu. Postupne zaraďuje prvky do skupín podľa symbolov na predošlých pozíciách. Počet krokov algoritmu je teda závislý od dĺžky triedených reťazcov.  
Zložitosť:  $T(n)=O((m+n)k)$ .

At the bottom of the window is the 'Ovládací panel' (Control Panel) with buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', 'Rýchlosť' (set to 6), 'Rád' (set to 2), 'Vlastné čísla zapnúť', and 'SK'.

Obr. 11: Radixsort

## Modul selectsort

Modul obsahuje vizualizáciu činnosti algoritmu triedenia výberom (selectsort). Jedná sa o jednoduchý triediaci algoritmus. V postupnosti sa nájde najmenší prvok a ten sa vymení s prvkom na prvej pozícii. Pozícia, kde bol vložený najmenší prvok sa už neprehľadáva a postup sa opakuje na zostávajúcej časti postupnosti. Vizualizácia činnosti algoritmu je zachytená na Obr. 12.

The screenshot shows the VizAlgo application window. At the top, there is a 'Výber algoritmu' dropdown menu set to 'selectsort'. Below it is the 'Pseudokód algoritmu' section with the following code:

```
for (i = 0; i < n-1; i++) {  
  for (j = i + 1; j < n; j++) {  
    if (x[i] > x[j]) {  
      temp = x[i];  
      x[i] = x[j];  
      x[j] = temp;  
    }  
  }  
}
```

The 'Vizualizácia algoritmu' section displays an array of numbers: 5, 51, 74, 65, 92, 55, 37, 87, 20, 6. The number 51 is highlighted in a green box, and 37 is highlighted in a red box. Red arrows labeled 'i' and 'j' point to the positions of 51 and 37 respectively. Below the array, it says 'Hodnoty premenných: i: 1 j: 6'. At the bottom, there is an 'Informácie' section with text: 'Metóda: SelectSort predstavuje algoritmus triedenia výberom. V postupnosti prvkov nájde najmenší prvok a ten zamení s prvkom na prvom mieste. Index, kde bol vložený najmenší prvok sa už neprehľadáva. Postup sa opakuje v zostávajúcej časti nezotriedenej postupnosti. Zložitosť:  $T(n) = O(n^2)$  porovnaní.' The control panel at the bottom includes buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', a speed slider set to 6, a 'Rád' dropdown set to 2, and a 'Vlastné čísla zapnúť' checkbox.

Obr. 12: Triedenie výberom

## Modul shellsort

Modul obsahuje vizualizáciu činnosti algoritmu Shellovho triedenia. Jedná sa o jednoduchý triediaci algoritmus pracujúci tak, že neradí prvky, ktoré sú priamo vedľa seba, ale prvky, medzi ktorými je v postupnosti určitá vzdialenosť. V každom kroku je potom vzdialenosť medzi prvkami zmenšená. Vizualizácia činnosti algoritmu s využitím stĺpcového zobrazenia triedených prvkov je zachytená na Obr. 13.

The screenshot shows the VizAlgo application window. At the top, there is a 'Panel vstupov' (Input Panel) with a dropdown menu for 'Výber algoritmu' (Algorithm Selection) set to 'shellsort' and a dropdown for 'Zobrazenie' (Display) set to 'Stĺpcové' (Columnar). Below this is the 'Pseudokód algoritmu' (Algorithm Pseudocode) section, which contains the following code:

```
for (int inc = n / 2;
    inc > 0; inc = (inc == 2 ? 1
    : (int) Math.round(inc / 2.2))) {
  for (int i = inc; i < n; i++) {
    int temp = x[i];
    for (int j = i; j >= inc
        && x[j - inc] > temp; j -= inc) {
      x[j] = x[j - inc];
      x[j - inc] = temp;
    }
  }
}
```

The bar chart, titled 'Vizualizácia algoritmu' (Algorithm Visualization), displays the current state of the array. The bars represent the values: 26, 66, 6, 62, 67, 94, 27, 23, 70, and 71. The bar for 66 is highlighted in green, and the bar for 27 is highlighted in red. Below the chart is an 'Informácie' (Information) section with a scrollable text area containing the following text:

priamo susedi. Shell sort funguje podobne. Základným rozdielom však je, že Shell sort využíva tzv. znižujúci sa prírastok. To znamená, že algoritmus neradí prvky, ktoré sú priamo vedľa seba, ale prvky, medzi ktorými je určitá medzera. V každom kroku je potom medzera medzi prvkami zmenšená. V okamihu keď sa veľkosť medzery zníži na 1 dôjde k zoradeniu susedných prvkov.

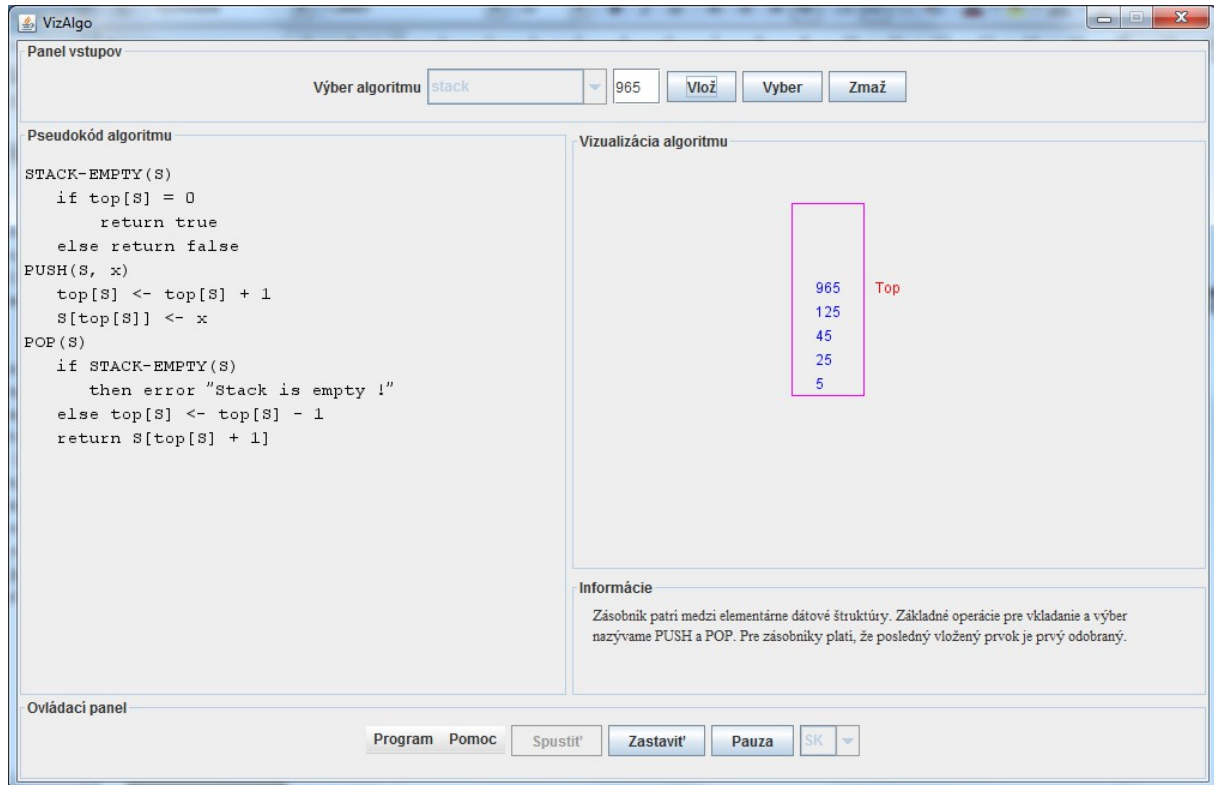
At the bottom, the 'Ovládací panel' (Control Panel) includes buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', a dropdown for 'Rýchlosť' (Speed) set to 6, a dropdown for 'Rád' (Order) set to 2, and a checkbox for 'Vlastné čísla zapnúť' (Enable custom numbers) which is checked. There is also a 'SK' button.

Obr. 13: Shellovo triedenie



## Modul stack

Modul obsahuje vizualizáciu činnosti údajovej štruktúry zásobník (stack) typu LIFO (Last In First Out). Vizualizácia je zachytená na Obr. 14.



Obr. 14: Údajová štruktúra zásobník

## Modul treeTraversal

Modul obsahuje vizualizáciu činnosti prechodu binárnym stromom stratégiami preorder, inorder a postorder. Vizualizácia na Obr. 15 zachytáva prechod stratégiou preorder.

The screenshot shows the VizAlgo application window. At the top, the 'Výber algoritmu' (Algorithm Selection) dropdown is set to 'treeTraversal' and the 'Preorder' dropdown is selected. The 'Pseudokód algoritmu' (Algorithm Pseudocode) section contains the following code:

```
preorder(node)
  if node == null then return
  visit(node)
  preorder(node.left)
  preorder(node.right)

inorder(node)
  if node == null then return
  inorder(node.left)
  visit(node)
  inorder(node.right)

postorder(node)
  if node == null then return
  postorder(node.left)
  postorder(node.right)
  visit(node)
```

The 'Vizualizácia algoritmu' (Algorithm Visualization) section displays a binary tree with nodes 39, 60, 66, 9, 52, 4, 55, 42, and 5. Node 55 is highlighted with a red border, indicating it is the current node being visited. Below the tree, the 'Preorder:' sequence is shown as: 60 39 52 4 42 5 55.

The 'Informácie' (Information) section provides the following descriptions:

- Preorder - najprv sa spracuje koreň, potom ľavý podstrom a nakoniec pravý podstrom.
- Inorder - najprv sa spracuje ľavý podstrom, potom koreň a nakoniec pravý podstrom.
- Postorder - najprv sa spracuje ľavý podstrom, potom pravý podstrom a nakoniec koreň.

The 'Ovládací panel' (Control Panel) at the bottom includes buttons for 'Program', 'Pomoc', 'Spustiť', 'Ďalej', 'Zastaviť', 'Prehrať', 'Rýchlosť' (set to 6), 'Rád' (set to 2), 'Vlastné čísla zapnúť', and 'SK'.

Obr. 15: Prechod binárnym stromom