Cvičení 3

R E K U R Z E   A   R E K U R Z I V N Í   D A T O V É   S T R U K T U R Y

V programovacích jazycích C / C++ a Java programově zpracujte následující úlohy:

1. Máte dán jednosměrně zřetězený uspořádaný seznam prvků obsahujících jako hodnoty libovolná celá čísla, přičemž deklarace prvku seznamu bude v jazyce C / C++ provedena následujícím způsobem:

```
typedef  struct element  *P_element;
typedef  struct element  {
            int value;
            P_element next;
         } ELEM;

P_element  L = NULL;
```

Zapište rekurzivní funkci, která zařadí nový prvek na příslušné místo v seznamu.

[2 body]

Návod: Prostřednictvím ukazatele na čelo seznamu porovnáte hodnotu nově zařazovaného prvku s hodnotou prvku na čele seznamu. Odpovídá-li výsledek porovnání podmínce uspořádání seznamu, zařadíte nový prvek na čelo seznamu. Neodpovídá-li, pak rekurzivně vyvoláte tutéž proceduru s hodnotou ukazatele na další prvek seznamu (na nové čelo o jeden prvek kratšího seznamu). Po nalezení odpovídajícího místa pro zařazení nového prvku tento zařadíte na čelo zbytku seznamu s tím, že nesmíte zapomenout vytvořit propojení (zřetězení) nově zařazovaného prvku s prohlédnutou částí seznamu. A pozor! Zařazení nového prvku na konec seznamu (jako poslední prvek) není výjimečný případ!

2. Mějte dán binární vyhledávací strom deklarovaný v jazyce C / C++ následujícím způsobem:

```
typedef  struct element  *P_node;
typedef  struct element  {
            int value;
            P_node left_son, right_son;
         }  NODE;

P_node   BVS = NULL;
```

Zapište funkci, která s využitím rekurze vypíše obsah BVS pootočený tak, aby nejlevější list stromu byl v pravé polovině prvé řádky a kořen stromu na začátku střední řádky obrazovky.

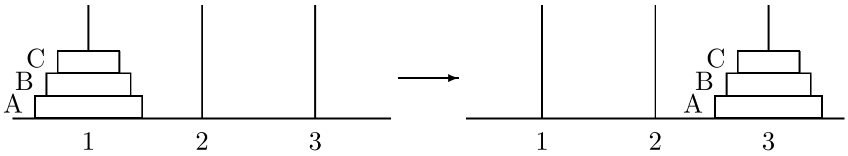
[2 body]

3. V podobě funkce `Vyber_Max` realizujte algoritmus metody vzestupného řazení prvků pole (tabulky) výběrem maximálního prvku (selection–sort) **s využitím rekurze**. Funkci procedury ověřte jednoduchým programem na vhodné, nepříliš velké množině testovacích dat.

[3 body]

4. Prohledávání stromu řešení úlohy

Mějte dánu klasickou úlohu přemísťování hanojských věží. V počátečním stavu úlohy je hanojská věž skládající se z N kotoučů o různých průměrech situována na levém kolíku, který označme 1 (viz obr. 1). Úkolem úlohy je přemístit jednotlivé kotouče na pravý kolík (označený 3) prostřednictvím středního kolíku 2 tak, že se smí vždy přemísťovat jen vrchní kotouč a žádný kotouč nesmí nikdy ležet na kotouči menšího průměru. Kotouče označme podle velikosti (průměru) od největšího po nejmenší písmeny A, B, C, ..., N (pod symbolem N si představme nejvyšší písmeno označující nejmenší kotouč, např. pro věž o třech kotoučích N = C − viz obr. 1).



Obr. 1: Úloha "Hanojské věže"

Libovolný stav úlohy popíšeme seznamem ["1" "2" "3"], kde symboly "1", "2", "3" představují seznamovou reprezentaci uložení kotoučů na kolíku 1, 2, či 3 v pořadí zdola nahoru. Nenachází-li se na kolíku žádný kotouč, bude seznam prázdný (reprezentován nil). Na obr. 1 vyobrazená úloha se pak dá symbolicky zapsat jako

$$H: \quad [\,[\,\mathrm{A\ B\ C}\,]\ \ \mathsf{nil}\ \ \mathsf{nil}\,] \quad \rightarrow \quad [\,\mathsf{nil}\ \ \mathsf{nil}\ \ [\,\mathrm{A\ B\ C}\,]\,]\ .$$

Elementární operací, definovanou nad takto formulovanou úlohou, budiž přemístění jednoho kotouče z jednoho kolíku na jiný. Např. přemístění nejmenšího kotouče z kolíku 1 na kolík 3 vyjádříme:

$$r_1: \quad [\,[\,\mathrm{A\ B\ C}\,]\ \ \mathsf{nil}\ \ \mathsf{nil}\,] \quad \rightarrow \quad [\,[\,\mathrm{A\ B}\,]\ \ \mathsf{nil}\ \ [\,\mathrm{C}\,]\,]\ .$$

Zadání:

1. Nakreslete pokud možno celý strom řešení úlohy pro věž o **třech** kotoučích.
2. Zakreslete do něj postup, kterým strom budete procházet při hledání žádaného (cílového) stavu prohledávací strategií
   a) do hloubky,
   b) do šířky.
3. Vyčíslete počet kroků (počet prohledaných uzlů stromu), které je třeba v obou případech vykonat pro nalezení cílového stavu (kotouče A, B, C na kolíku 3).

[à 1 bod]