

# Slovník

*semestrální práce* ***D S A***

**Aleksandr Shabelnikov**

# Zadání

- Napište program realizující slovník, ve kterém českému slovu (bez diakritiky) odpovídá jedno nebo několik anglických slov. Počáteční obsah slovníku bude dán textovým souborem. Program pak umožňuje slovník doplňovat, opravovat a v něm hledat.

# Operace

- najít anglické překlady k zadanému slovu
- zrušit záznam pro zadané slovo
- vložit nový překlad nebo i nové slovo k překladu
- vypsát slovník

# Realizace

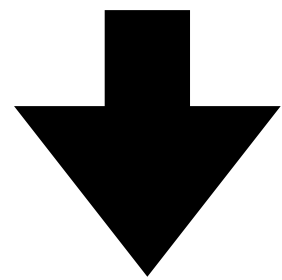
*a.k.a. co umí můj slovník*

```
-----  
| Dictionary |  
-----  
  
Semestrál project for D S A  
2022, Aleksandr Shabelnikov, VSPJ  
  
Loading data from file  
Opening file /Users/mrmidi/CLionProjects/SEM/cs-en.csv  
Loaded 2000 words  
Time: 0.002815  
1. Add new word  
2. Search czech word  
3. Search english word  
4. Delete word  
5. Print dictionary  
6. Make new file with cs-en dictionary  
7. Make new file with en-cs dictionary  
8. Exit  
Enter your choice:
```

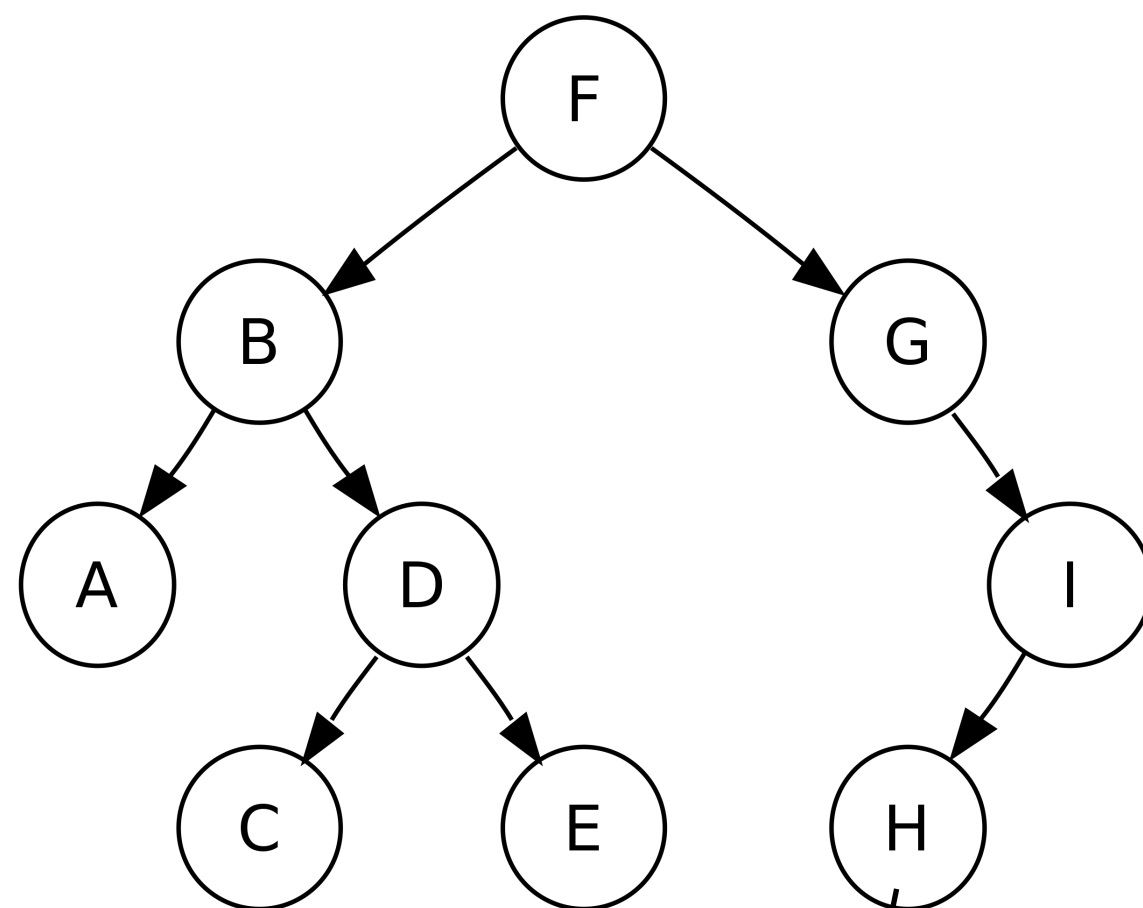
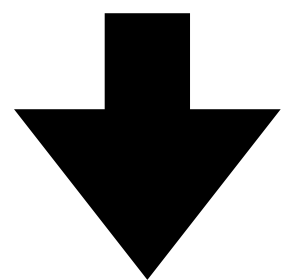
# Realizace

- Datová struktura: BST (*nevyvážený*)
- Druhá datava struktura: BST!
- Hašování pomocí C++ `std::hash`

CS slovo



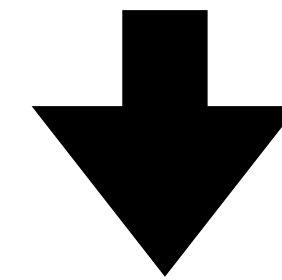
Hash



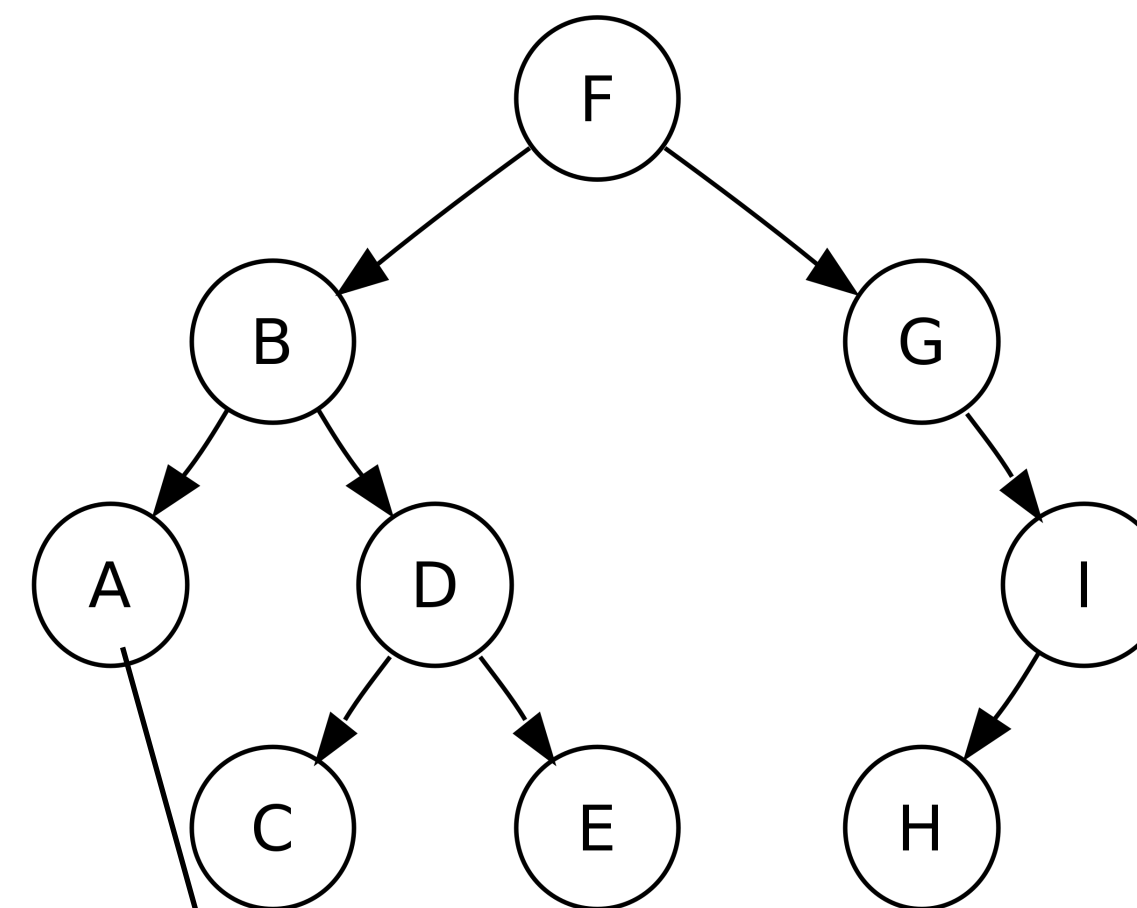
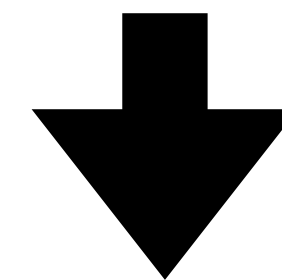
BST Node

Cs slovo (string word)  
EN překlad

EN slovo

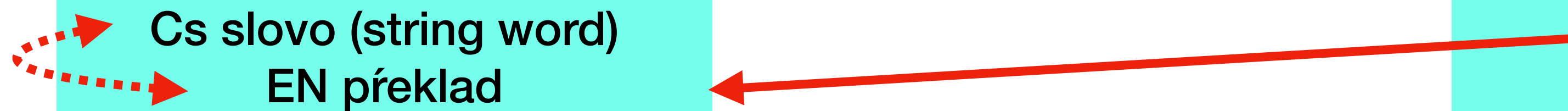


Hash



BST Node

**POINTER**  
na BST Node



# Realizace

## *Node.h*

```
class Node {
public:
    size_t key;
    std::string word;
    std::string definition;
    Node *pLeft;
    Node *pRight;
    Node(size_t key, std::string word, std::string definition)
    {
        this->key = key;
        this->word = word;
        this->definition = definition;
        pLeft = NULL;
        pRight = NULL;
    }
};
```

# Realizace

## *Node\_EN.h*

```
class Node_EN {
public:
    size_t key;
    Node_EN *pLeft;
    Node_EN *pRight;
    Node *pNode;
    Node_EN(size_t key, Node &pNode)
    {
        this->key = key;
        this->pNode = &pNode; // reference to the node in another language
        pLeft = NULL;
        pRight = NULL;
    }
};
```

# Realizace

## *BST.h & BST\_En.h*

```
public:
    BST(); // constructor
    void insert(size_t key, std::string word, std::string definition);
    Node * insertWithReturn(size_t key, std::string word, std::string definition);
    void remove(size_t key);
    Node* search(size_t key);
    void print();
    void print(Node *pNode);
    void clear();
    void clear(Node *pNode);
    void getRoot();
    Node *getSuccessor(Node *pDelNode);
    void makeFile();
    void makeFile(Node *pNode, std::ofstream &file);
    void makeFileEn();
    void makeFileEn(Node *pNode, std::ofstream &file);
```

```
public:
    BST_EN(); // constructor
    void insert(size_t key, Node pNode);
    void remove(size_t key);
    Node_EN* search(size_t key);
    void print();
    void print(Node_EN *pNode);
    void clear();
    void clear(Node_EN *pNode);
    void getRoot();
    void insert(size_t key, Node *pNode);
    Node_EN *getSuccessor(Node_EN *pNode);
```



# Poznamki

- Proč nevyvážený
- Proč 2 stromy
- Co vylepšit (kolizie)

**Děkují za pozornost**