

Vyhledávání

Programovací techniky

doc. Ing. Jiří Rybička, Dr.
ústav informatiky
PEF MENDELU v Brně
rybicka@mendelu.cz

Princip vyhledávání

- Jedna z nejfrekventovanějších úloh

- Jedna z nejfrekventovanějších úloh
- Uložení dat a jejich efektivní vyhledání

- Jedna z nejfrekventovanějších úloh
- Uložení dat a jejich efektivní vyhledání
- Lze využít prakticky jakoukoliv strukturu

- Jedna z nejfrekventovanějších úloh
- Uložení dat a jejich efektivní vyhledání
- Lze využít prakticky jakoukoliv strukturu
- Důležitá je efektivnost v dané aplikaci

- Jedna z nejfrekventovanějších úloh
- Uložení dat a jejich efektivní vyhledání
- Lze využít prakticky jakoukoliv strukturu
- Důležitá je efektivnost v dané aplikaci
- Datová složka + klíč (jednoduchý typ, řetězec)

- Jedna z nejfrekventovanějších úloh
- Uložení dat a jejich efektivní vyhledání
- Lze využít prakticky jakoukoliv strukturu
- Důležitá je efektivnost v dané aplikaci
- Datová složka + klíč (jednoduchý typ, řetězec)
- Ideální případ: Klíč je jednoznačný v rámci dané struktury

- Jedna z nejfrekventovanějších úloh
- Uložení dat a jejich efektivní vyhledání
- Lze využít prakticky jakoukoliv strukturu
- Důležitá je efektivnost v dané aplikaci
- Datová složka + klíč (jednoduchý typ, řetězec)
- Ideální případ: Klíč je jednoznačný v rámci dané struktury
- Datové složky se stejným klíčem = synonyma

- Jedna z nejméně používaných úloh
- Uložení dat a jejich efektivní vyhledání
- Lze využít prakticky jakoukoliv strukturu
- Důležitá je efektivnost v dané aplikaci
- Datová složka + klíč (jednoduchý typ, řetězec)
- Ideální případ: Klíč je jednoznačný v rámci dané struktury
- Datové složky se stejným klíčem = synonyma
- Co nejrychlejší přístup ke složkám – záznamům

Realizace vyhledávání

- Lineární struktury, nelineární struktury

- Lineární struktury, nelineární struktury
- Soubory: sekvenční přístup; uspořádané soubory: sekvenční přístup, půlení intervalu

- Lineární struktury, nelineární struktury
- Soubory: sekvenční přístup; uspořádané soubory: sekvenční přístup, půlení intervalu
- Lineární seznamy: sekvenční přístup; uspořádané seznamy: sekvenční přístup

- Lineární struktury, nelineární struktury
- Soubory: sekvenční přístup; uspořádané soubory: sekvenční přístup, půlení intervalu
- Lineární seznamy: sekvenční přístup; uspořádané seznamy: sekvenční přístup
- Stromy: stromový průchod; usp. stromy: stromové hledání

- Lineární struktury, nelineární struktury
- Soubory: sekvenční přístup; uspořádané soubory: sekvenční přístup, půlení intervalu
- Lineární seznamy: sekvenční přístup; uspořádané seznamy: sekvenční přístup
- Stromy: stromový průchod; usp. stromy: stromové hledání
- Pole: sekvenční přístup; usp. pole: sekvenční přístup, indexace

- Lineární struktury, nelineární struktury
- Soubory: sekvenční přístup; uspořádané soubory: sekvenční přístup, půlení intervalu
- Lineární seznamy: sekvenční přístup; uspořádané seznamy: sekvenční přístup
- Stromy: stromový průchod; usp. stromy: stromové hledání
- Pole: sekvenční přístup; usp. pole: sekvenční přístup, indexace
- Dělení podle časové složitosti: lineární, logaritmické, konstantní hledání

- Časová složitost lineární = nejhorší

- Časová složitost lineární = nejhorší
- Všude dále v příkladech platí: pole P , s N složkami, rozměr pole může být $M \geq N$

- Časová složitost lineární = nejhorší
- Všude dále v příkladech platí: pole P, s N složkami, rozměr pole může být $M \geq N$
- Sekvenční hledání:

```
I:=1;  
while (I<=N) and (C<>P[I]) do inc(I);  
Nalezeno:=I<=N;
```

Sekvenční hledání se záložkou

Sekvenční hledání se zářátkou

- Časová složitost stejná jako u předchozí metody

Sekvenční hledání se zářátkou

- Časová složitost stejná jako u předchozí metody
- Na zářátku musí být volné místo

Sekvenční hledání se zářížkou

- Časová složitost stejná jako u předchozí metody
- Na zářížku musí být volné místo
- Vložení zářížky s konstantní složitostí

```
I:=1; P[N+1]:=C;  
while C<>P[I] do inc(I);  
Nalezeno:=I<=N;
```

Sekvenční hledání v uspořádaném poli

Sekvenční hledání v uspořádaném poli

- Časová složitost opět lineární = nejhorší

Sekvenční hledání v uspořádaném poli

- Časová složitost opět lineární = nejhorší
- Velké plýtvání – nevyužívá se seřazené struktury

```
I:=1;  
while (I<=N) and (C>P[I]) do inc(I);  
Nalezeno:=(I<=N) and (C=P[I]);
```


- Rozmezí prvků pole se rozdělí na poloviny

- Rozmezí prvků pole se rozdělí na poloviny
- U středového prvku provedeme porovnání s hledanou hodnotou

- Rozmezí prvků pole se rozdělí na poloviny
- U středového prvku provedeme porovnání s hledanou hodnotou
- Na základě testu pak stejným způsobem pracujeme s levou, nebo s pravou polovinou daného rozmezí

- Rozmezí prvků pole se rozdělí na poloviny
- U středového prvku provedeme porovnání s hledanou hodnotou
- Na základě testu pak stejným způsobem pracujeme s levou, nebo s pravou polovinou daného rozmezí
- Konec neúspěšného hledání: rozmezí prvků je tvořeno jediným prvkem

- Binární vyhledávací strom, uspořádání: $ls \leq o < ps$

- Binární vyhledávací strom, uspořádání: $ls \leq o \leq ps$
- Implementace dynamickou strukturou:

```
pom:=koren;  
while (pom<>nil) and (pom^.klic<>C) do  
    if pom^.klic<C then pom:=Pom^.ps  
        else pom:=pom^.ls;  
Nalezeno:=Pom<>nil;
```


- Nejrychlejší možná metoda vyhledání

- Nejrychlejší možná metoda vyhledání
- Silné omezení na datové hodnoty

- Nejrychlejší možná metoda vyhledání
- Silné omezení na datové hodnoty
- Implementace množinou

- Nejrychlejší možná metoda vyhledání
- Silné omezení na datové hodnoty
- Implementace množinou
- Implementace logickým polem

- Nejrychlejší možná metoda vyhledání
- Silné omezení na datové hodnoty
- Implementace množinou
- Implementace logickým polem
- Implementace celočíselným polem

- Kombinace indexace a sekvenčního hledání

- Kombinace indexace a sekvenčního hledání
- Jednoduchá implementace

- Kombinace indexace a sekvenčního hledání
- Jednoduchá implementace
- Rozptylování: funkce pracující v konstantním čase

- Kombinace indexace a sekvenčního hledání
- Jednoduchá implementace
- Rozptylování: funkce pracující v konstantním čase
- Omezení počtu synonym: konstantní složitost, jinak lineární

```
const Max=211;
type TypKlic = longint;
  Indexy = 1..Max;
  Data = record
    Klic: TypKlic;
    Udaje: pointer;
  end;
UkClen = ^Clen;
Clen = record
  D: Data;
  N: UkClen;
end;
ZaklPole = array [Indexy] of UkClen;
HashTable = ^ZaklPole;
```



```
function Hash(K: TypKlic): Indexy;  
  begin Hash:=K mod Max + 1;  
  end;  
  
procedure Init(var H: HashTable);  
  var I:Indexy;  
  begin new(H);  
    for I:=1 to Max do H^[I]:=nil;  
  end;
```

```
procedure HVloz(var H:HashTable; El: Data);  
  var I:Indexy; Pom: UkClen;  
  begin I:=Hash(El.Klic);  
        new(pom);  
        Pom^.D:=El;  
        Pom^.N:=H^[I];  
        H^[I]:=Pom;  
  
  end;
```

```
function HHledej (H: HashTable;  
                  K: TypKlic): UkClen;  
var I: Indexy; Pom: UkClen;  
begin I:=Hash (K) ;  
      Pom:=H^ [I] ;  
      while (Pom<>nil) and (Pom^.D.Klic<>K)  
          do Pom:=Pom^.N;  
      HHledej:=Pom  
end;
```