

Системне програмування I

*Пустоваров В. І., НТУУ"КПІ",
м. Київ vpustovarov@ukr.net*

Лекція 1/7

***Відношення порядку, впорядкування таблиць та
використання двійкового пошуку для вибірки
даних***

- 1. Таблиці, їх поля та рядки*
- 2. Пошук в таблицях*
- 3. Впорядкування таблиць за лексикографічним
порядком ключів та двійковий пошук*

Відношення порядку

Впорядкування таблиць за ключовими полями стає можливим лише у випадку впорядкування кодів кожного з цих ключових полів. Відношення порядку встановлюються для даних з одновимірними множинами визначення (доменами) значень. Всі числові і перенумеровані типи даних, в тому числі і полів мають визначені відношення порядку. Тому набори операцій мови C/C++ "==" та "!=" , які створюються за умовчанням, необхідно розширити додатковими операціями відношень мови C "<", "<=", ">=" та ">", в тому числі з урахуванням полів з покажчиками

Для визначення відношення порядку ключа, що складається з декількох полів або багатокomпонентних типів, необхідно, щоб кожний з типів полів мав власне відношення порядку, і щоб узагальнююче відношення будувалося за допомогою монотонних функцій. Поля текстових типів впорядковуються за лексикографічним (словниковим) порядком. Для роботи з літерами кирилиці та інших національних алфавітів доводиться використовувати функції алфавітного впорядкування літер.

Узагальнення відношень порядку

В найбільш загальному випадку ключі таблиць можуть включати декілька полів, в тому числі і поля показчиків, адрес або посилань.

Для визначення відношення порядку ключа, що складається з декількох полів або багатокomпонентних типів даних в ключах, необхідно, щоб кожний з типів полів мав власне відношення порядку. Тоді узагальнююче відношення можна побудувати за допомогою монотонних функцій.

При роботі з показниками, адресами або посиланнями з початку треба одержати доступ до даних полів, а лише потім використовувати функції порівняння окремих полів.

Функції порівняння

// порівняння рядків за відношенням нерівності

```
int neqKey(struct recrd* el, struct keyStr kArg)
```

```
    {return (strcmp(el->key.str, kArg.str))||
```

```
        el->key.nMod != kArg.nMod);
```

```
}
```

// порівняння структур за лексикографічним (словарним) порядком

```
int cmpStr(unsigned char* s1, unsigned char* s2)
```

```
{unsigned n=0;
```

```
    while (s1[n]==s2[n]&& s1[n]!=0) n++;
```

```
    return s1[n]-s2[n];
```

```
}
```

// порівняння рядків за відношенням порядку двох полів

```
int cmpKey(struct recrd* el, struct keyStr kArg)
```

```
{int i=cmpStr((unsigned char*)el->key.str,
```

```
              (unsigned char*)kArg.str);
```

```
    if(i)return i;
```

```
    return el->key.nMod - kArg.nMod;
```

```
}
```

Приклад порівняння рядків на мові Асемблера

Процедура порівняння рядків за відношенням порядку на мові С та з вставкою на мові Асемблер має вигляд:

```
char cmpSts(char *s0, char *s1)
{ _asm{
    push esi
    push edi
    push ecx
    xor     eax,eax    ; очистка акумулятора
    xor     ecx,ecx    ; очистка лічильника
    mov     edi,s1
    mov     esi,s0
lb: lodsb          ; завантаження чергового знака 1-го рядка
   scasb          ; порівняння з черговим знаком 2-го рядка
   jne     lf      ; вихід при неспівпадінні
   cmp     eax,0    ; порівняння з кінцем 1-го рядка
   je      lf      ; перехід за умови кінця
   loop    lb       ; на обробку наступного знака
lf: sub     al,[edi-1] ; формування зваженої умови порядку
   pop     ecx
   pop     edi
   pop     esi
}}
```

Двійковий пошук

// вибірка за двійковим пошуком

```
struct recrd*selBin (struct keyStr kArg,// ключ аргументу пошуку
struct recrd*tb, // адреса початку таблиці
int ln) // кількість елементів таблиці
{int i, nD=-1, nU=ln, n=(nD+nU)>>1;
 while (i=cmpKey(&tb[n],kArg))
 {
  if (i>0)nU=n; else nD=n;
  n=(nD+nU)>>1;
  if (n==nD) return NULL;
 }
 return &tb[n];
}
```

// вилучення за двійковим пошуком

```
struct recrd*delLin(struct recrd*pElm,
                   struct recrd*tb, int *pQnElm)
{struct recrd*pEl=selBin(pElm->key, tb, *pQnElm);
 if(pEl)pEl->_del=-1;
 return pEl;
}
```

Підсумки

- До найбільш поширених базових методів роботи з таблицями належать пошук за прямою адресою, лінійний пошук та двійковий пошук у впорядкованих таблицях.
- Процедури корекцій таблиць спираються на процедури лінійного та двійкового пошуку та порівняння ключів.
- Процедури порівняння простих і комплексних ключів спираються на перевірку відповідних відношень порядку.

Поточна контрольна робота

- Виконується за виданими варіантами.