# **Національний технічний університет України**

# **«Київський політехнічний інститут»**

# **Факультет інформатики та обчислювальної техніки**

# **Кафедра обчислювальної техніки**

# **Лабораторна робота №2**

### **з програмування**

виконав студент групи ІО-01

Редько Олександр

Номер залікової книжки: 103

**Тема:** Списки в мові програмування Pascal.

**Мета:** Здобуття навичок у створенні процедур та функцій для роботи зі списками. Розробка діалогового інтерфейсу користувача та функціонального наповнення. Розробка засобів обробки та редагування даних.

**Завдання:**

Створити модуль, який складається з власного типу (див. нижче), процедур (функцій) створення списку, додавання записів до списку, видалення записів зі списку, перегляду записів у списку та перегляду усіх записів списку у вигляді таблиці. Створити програму, яка має діалоговий інтерфейс , що реалізує визначені варіантом та завданням процедури (функції).

1. Тип даних (): – інформація про літак.
2. Тип списку (): двозв’язний кільцевий.
3. Тип створення списку (): додавання *n* записів до списку. Кожен наступній запис додається в кінець списку.
4. Типи додавання записів (): додавання 1 запису в кінець списку. Додавання *n* записів в початок списку.
5. Типи видалення записів (): видалення 1 запису з вказаної позиції списку. Видалення *n* записів з початку списку.
6. Типи перегляду записів у файлі ): перегляд першого запису.

**Лістинг модуля:**

**Unit** Lab2mod;

**Interface**

**type**

TPlane = ^Plane;

Plane = **record**

manuf: string[15];

model: string[10];

length: integer;

height: integer;

wingspan: integer;

prev: TPlane;

next: TPlane;

**end**;

**function** input: TPlane;

**procedure** addToEnd(**var** ring, rec: TPlane; n: integer);

**procedure** addToStart(**var** ring, rec: TPlane);

**procedure** create(**var** ring: TPlane; n: integer);

**function** view(ring: TPlane; n: integer): TPlane;

**procedure** viewFirst(ring: TPlane);

**function** size(ring: TPlane): integer;

**procedure** deleteOne(**var** ring: TPlane);

**procedure** deletePos(**var** ring: TPlane; n: integer);

**procedure** deleteFromStart(**var** ring: TPlane; n: integer);

**procedure** viewAll(ring: TPlane);

**Implementation**

**function** input: TPlane;

**var**

rec: TPlane;

**begin**

**new**(rec);

write('Enter manufacturer: ');

readln(rec^.manuf);

{write('Enter model: ');

readln(rec^.model);

write('Enter length: ');

readln(rec^.length);

write('Enter height: ');

readln(rec^.height);

write('Enter wingspan: ');

readln(rec^.wingspan);}

writeln;

input := rec;

**end**;

**procedure** addToEnd(**var** ring, rec: TPlane; n: integer);

**var**

tmp: TPlane;

**begin**

**if** ring = nil **then**

**begin**

ring := rec;

rec^.next := rec;

rec^.prev := rec;

**end**

**else**

**begin**

tmp := ring;

rec^.next := tmp;

rec^.prev := tmp^.prev;

tmp^.prev^.next := rec;

tmp^.prev := rec;

**end**;

**end**;

**procedure** addToStart(**var** ring, rec: TPlane);

**var**

tmp: TPlane;

**begin**

tmp := ring^.prev;

rec^.next := tmp^.next;

rec^.prev := tmp;

tmp^.next^.prev := rec;

tmp^.next := rec;

ring := rec;

**end**;

**procedure** create(**var** ring: TPlane; n: integer);

**var**

i: integer;

rec: TPlane;

**begin**

ring := nil;

**for** i := n **downto** 1 **do**

**begin**

rec := input;

addToEnd(ring, rec, n);

**end**;

**end**;

**function** view(ring: TPlane; n: integer): TPlane;

**var**

tmp: TPlane;

**begin**

tmp := ring;

**while** n > 0 **do**

**begin**

tmp := tmp^.next;

n := n - 1;

**end**;

view := tmp;

**end**;

**function** size(ring: TPlane): integer;

**var**

i: integer;

tmp: TPlane;

**begin**

i := 0;

**if** ring <> nil **then**

**begin**

tmp := ring;

i := 1;

**while** tmp^.next <> ring **do**

**begin**

tmp := tmp^.next;

inc(i);

**end**;

**end**;

size := i;

**end**;

**procedure** deleteOne(**var** ring: TPlane);

**var**

tmp: TPlane;

**begin**

tmp := ring;

**if** ring <> nil **then**

**begin**

tmp^.next^.prev := tmp^.prev;

tmp^.prev^.next := tmp^.next;

ring := tmp^.next;

dispose(tmp);

**end**;

**end**;

**procedure** deletePos(**var** ring: TPlane; n: integer);

**var**

tmp: TPlane;

**begin**

tmp := ring;

**if** ring <> nil **then**

**begin**

**if** n <> 1 **then**

**while** n > 0 **do**

**begin**

tmp := tmp^.next;

n := n - 1;

**end**;

tmp^.next^.prev := tmp^.prev;

tmp^.prev^.next := tmp^.next;

**if** n = 1 **then**

ring := tmp^.next;

dispose(tmp);

**end**

**end**;

**procedure** deleteFromStart(**var** ring: TPlane; n: integer);

**var**

i, siz: integer;

**begin**

siz := size(ring);

**if** n <> siz **then**

**for** i := 1 **to** n **do**

deleteOne(ring)

**else**

ring := nil;

**end**;

**procedure** viewFirst(ring: TPlane);

**var**

rec: TPlane;

**begin**

rec := view(ring, 0);

writeln(rec^.manuf:8,''{, rec^.model:8, ' ',rec^.length:8, '', rec^.height:8, '', rec^.wingspan:8});

writeln;

**end**;

**procedure** viewAll(ring: TPlane);

**var**

i: integer;

rec: TPlane;

**begin**

**for** i := 1 **to** size(ring) **do**

**begin**

rec := view(ring, i - 1);

writeln(rec^.manuf:8,''{, rec^.model:8, ' ',rec^.length:8, '', rec^.height:8, '', rec^.wingspan:8});

writeln;

**end**;

**end**;

**begin**

**end**.

**Лістинг основної програми:**

**Program** Lab2;

**uses** crt, lab2mod;

**var**

ring, rec: TPlane;

n, i, siz, number: integer;

flag: boolean;

**begin**

siz := size(ring);

**repeat**

writeln('Choose operation');

writeln('1. Create Ring');

writeln('2. Add 1 to end');

writeln('3. Add n to start');

writeln('4. Delete 1 from position');

writeln('5. Delete n from start');

writeln('6. Show first');

writeln('7. Show all');

writeln('0. Exit');

readln(number);

**case** number **of**

1:

**begin**

clrscr;

**if** ring = nil **then**

**repeat**

writeln('number of records: ');

readln(n);

flag := false;

**if** n < 1 **then**

**begin**

writeln('n can not be negative or zero!');

flag := false;

**end**

**else**

**begin**

create(ring, n);

writeln(size(ring));

flag := true;

**end**;

**until** flag = true

**else**

writeln('Ring is already created!');

writeln;

**end**;

2:

**begin**

clrscr;

**if** ring <> nil **then**

**begin**

writeln('record to add to end: ');

rec := input;

addToEnd(ring, rec, 1);

**end**

**else**

writeln('First, create ring!');

writeln;

**end**;

3:

**begin**

clrscr;

**if** ring <> nil **then**

**begin**

writeln('number of records to add to start');

readln(n);

**for** i := n **downto** 1 **do**

**begin**

rec := input;

addToStart(ring, rec);

**end**;

**end**

**else**

writeln('First, create ring!');

writeln;

**end**;

4:

**begin**

clrscr;

siz := size(ring);

**if** ring <> nil **then**

**begin**

**repeat**

writeln('position: ');

//readln(n);

n := 1;

**if** n < 0 **then**

**begin**

writeln('Position can not be negative');

flag := false;

**end**

**else**

**begin**

**if** n > siz **then**

**begin**

writeln('Position exists');

flag := false;

**end**

**else**

**begin**

deletePos(ring, n);

writeln('OK');

flag := true;

**end**;

**end**

**until** flag = true;

**end**

**else**

writeln('First, create ring!');

writeln;

**end**;

5:

**begin**

clrscr;

siz := size(ring);

**if** ring <> nil **then**

**repeat**

writeln('number of records to delete');

readln(n);

**if** n < 0 **then**

**begin**

writeln('number can not be negative');

flag := false;

**end**

**else**

**if** n <= siz **then**

**begin**

deleteFromStart(ring, n);

writeln('OK');

flag := true;

**end**

**else**

**begin**

writeln('no such records');

flag := false

**end**;

**until** flag = true

**else**

writeln('First, create ring!');

writeln;

**end**;

6:

**begin**

clrscr;

**if** ring <> nil **then**

**begin**

writeln('First record:');

viewFirst(ring);

writeln;

**end**

**else**

writeln('Ring is empty');

writeln;

**end**;

7:

**begin**

clrscr;

**if** ring <> nil **then**

**begin**

writeln(' Ring: ');

viewAll(ring);

writeln;

**end**

**else**

writeln('Ring is empty');

writeln;

**end**;

**end**;

**until** number = 0;

**end**.

**Висновок:** Навчився працювати зі списками у мові програмування Pascal