**Семинар Списочные структуры данных 3. Корректировка списков.**

К корректировке списков относятся:

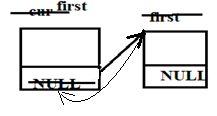
* Изменение элементов списка.
* Добавление элементов в список.
* Удаление элементов из списка.
* Сортировка списков.

1. **Корректировка однонаправленных списков. Добавление и удаление элементов. Основные приемы.**
2. **Добавление элементов в список.**

**cur=newelem;// Cоздание нового элемента**

**cur->num=k;**

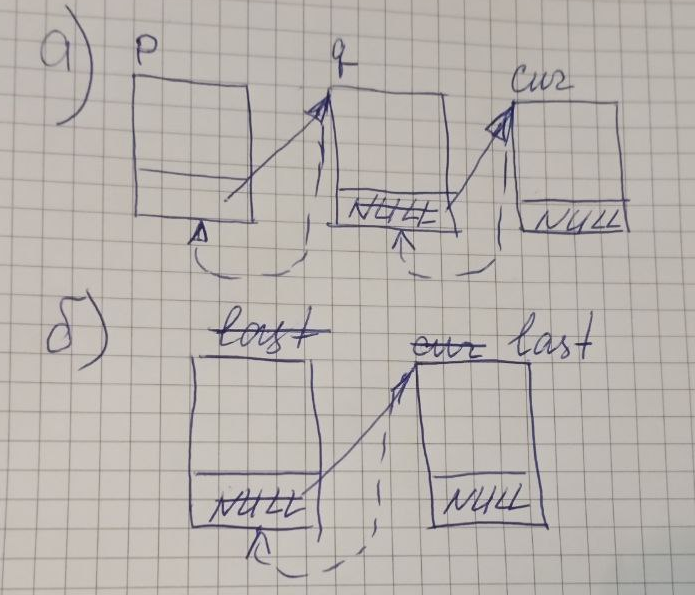
**cur->next=NULL;**

1. ***Добавление в начало списка***

cur->next=first;

first=cur

1. ***Добавление в конец списка***

А) last нет

q=first;

while(q->next!=NULL)

q=q->next;

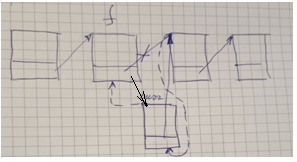
q->next=cur;

Б) last есть

last->next=cur;

last=cur;

1. ***Добавление после элемента с***

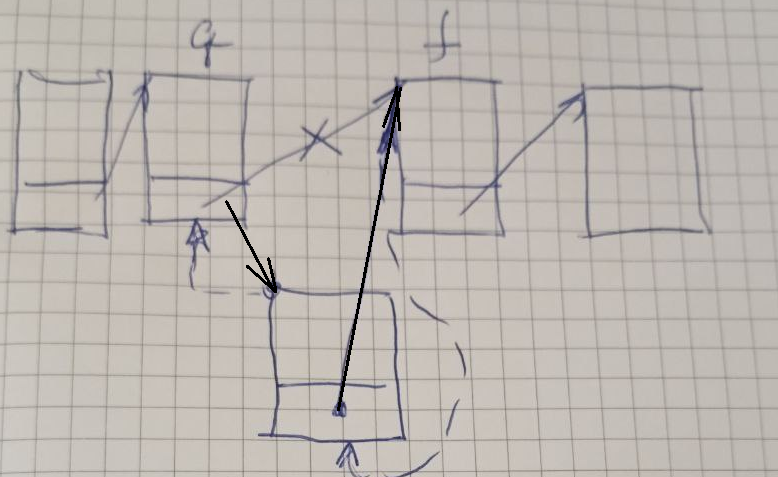
***адресом f.***

cur->next=f->next;

f->next=cur;

1. ***Добавление перед элементом с***

***адресом f***

q=first;

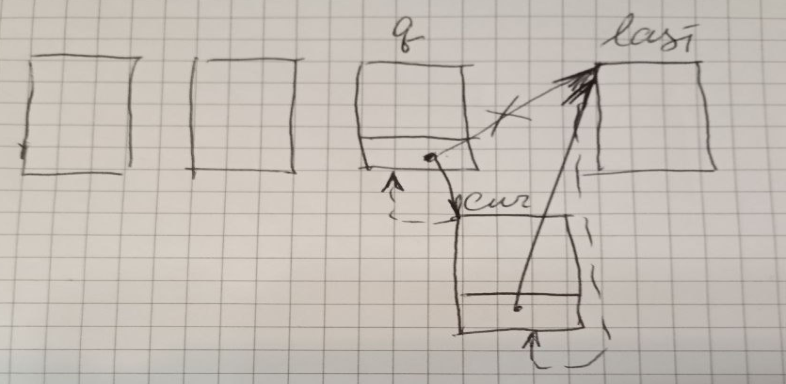
while(q->next!=f)

q=q->next;

q-next=cur;

cur->next=f;

1. ***Добавление перед последним***

*** элементом***

1. last есть

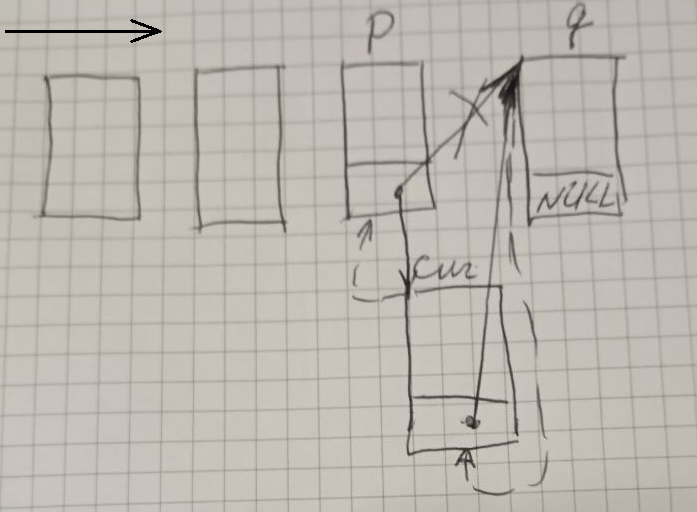
q=first;

while(q->next!=last)

q=q->next;

q-next=cur;

cur->next=last;

Б) last нет

p=first;

q=first;

while(q->next!=NULL)

{ p=q;

q=q->next;

}

p->next=cur;

cur->next=q;

1. **Удаление элементов из списка**
2. Изображение выглядит как доска, текст

   Автоматически созданное описание***Удаление первого элемента***

**cur=first;**

first=first->next;

// first=cur->next;

delete cur;

1. ***Удаление последнего элемента***

Изображение выглядит как текст

Автоматически созданное описание

А) last есть

q=first;

while(q->next!=last)

q=q->next;

cur=last;// delete last;

q->next=NULL;

last=q;

delete cur;

Б) last нет

f=first;

q=f->next;

while(q->next!=NULL)

{ f=q;

q=q->next;

}

f->next=NULL;

delete q;

1. ***Изображение выглядит как доска, текст

   Автоматически созданное описание Удаление элемента с адресом f.***

q=first;

while(q->next!=f)

q=q->next;

q->next=f->next;

delete f;

1. ***Удаление после элемента с адресом f.***

cur=f->next;

f->next=f->next->next;

// f->next=cur->next;

delete cur;

1. ***Удаление элемента перед элементом с адресом f***

Изображение выглядит как текст, доска

Автоматически созданное описаниеp=first;

q=p->next;

while(q->next!=f)

{ p=q;

q=q->next;

}

p->next=f;

delete q;

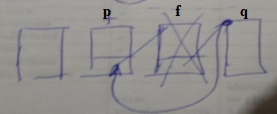
1. **Удаление перед последним**

p=first;

f=p->next;

q=f->next;

while(q->next!=NULL)

{p=f;

f=q;

q=q->next;

}

p->next=q;

delete f;

**Пример 1**

**Из чисел, полученных с помощью датчика случайныхчисел построить список. Вывести его на экран. Удалить из списка все числа, кратные введенному с клавиатуры. Вывести новыйсписок.**

**Как показал анализ, в списке могут быть следующие ситуации:**

* **Все элементы кратные введенному.**
* **В начале или середине списка есть подряд идущие несколько элементов, кратных введенному.**
* **Элементов кратных нет.**

**Поэтому, сначала организуем цикл проверки кратности первого элемента введенному. Пока такие элементы есть, мы удаляем их, как первый элемент.**

**После этого, если список еще есть, идем по списку, находим кратные элементы – и удаляем их по соответствующим правилам, приведенным выше.**

**#include<stdio.h>**

**#include<string.h>**

**#include<stdlib.h>**

**#include<time.h>**

**#include <conio.h>**

**// Элемент списка**

**struct rec{**

**int num;**

**rec \*p;};**

**int main()**

**{rec \*first,\*q,\*p;**

**int i,n;**

**srand( (unsigned)time( NULL ) );**

**n=rand()%30+10;// определение количества элементов**

**// количество элементов будет в диапазоне от 10 до 39**

**first=new rec;**

**first->p=NULL;**

**first->num=rand()%100+1;// чтобы не было элементов =0**

**// создаем первый элемент**

**p=first;**

**// формирование списка**

**for (i=1;i<n;i++){**

**q=new rec;**

**q->num=rand()/100+1;//чтобы не было элементов =0**

**// создаем новый элемент**

**q->p=NULL;**

**p->p=q;**

**p=q;**

**}**

**puts("Spisok");**

**q=first;**

**while(q!=NULL){**

**printf("%5d",q->num);**

**q=q->p;}**

**printf("\n");**

**puts("input value for udal");**

**scanf("%d",&n);**

**puts(" Find Elements for udal ");**

**// сначала проверяем первый элемент, пока он кратер n**

**// и удаляем его**

**while((first!=NULL)&&(first->num%n==0))**

**{ p=first;**

**first=first->p;**

**printf("%5d",q->num);**

**delete(p);**

**}**

**if (first==NULL) // удалены все элементы**

**puts("\nSpisok Empty");**

**else**

**{ // если список еще есть**

**p=first;**

**q=first;**

**do{**

**if(q->num%n==0) // находим кратный и удаляем**

**{ p->p=q->p;**

**printf("%5d",q->num);**

**delete(q);**

**q=p->p;**

**}**

**else**

**{ p=q;**

**q=q->p;**

**}**

**} while (q!=NULL);**

**puts("\nNew Spisok");**

**q=first;**

**while(q!=NULL)**

**{**

**printf("%5d",q->num);**

**q=q->p;**

**}**

**printf("\n");**

**}**

**q = first;**

**puts("Delete spisok");**

**while (q!=NULL)**

**{**

**p = q;**

**q = q->p;**

**delete p;**

**}**

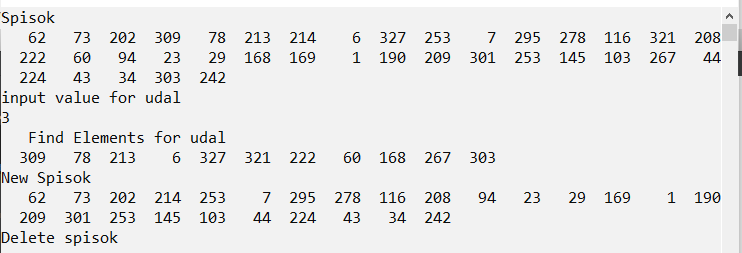
**first=p=NULL;**

**getch();**

**return 0;**

**}**

**Пример работы программы**

****

Другие примеры по удалению элементов приведены в семинарах 1 и 2

1. **Сортировка списка**

Для сортировки списков можно использовать любой метод сортировки. Однако на практике наиболее часто применяют сортировку готового списка методом пузырька или сортировку при вводе методом вставки.

**Пример 2**

**С клавиатуры вводится символьная строка, содержащая целые числа, записанные через пробел. Из чисел строки построить список, упорядоченный по возрастанию его элементов**

Для сортировки списка воспользуемся сортировкой при вводе (метод вставки). Для реализации используем однонаправленный список.

**#include<string.h>**

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<conio.h>**

**// Описание элемента списка**

**struct elem {**

**int num;**

**elem \* next;**

**};**

**int main()**

**{ elem \* first, \*last,\*p,\*q,\*n;**

**char s[80]; // для ввода чисел**

**char sl[10];// слово для одного числа**

**// и его преобразовани в число**

**int j,i,k,m;int fl;**

**puts("Input string with number");**

**gets(*s*);**

**first=NULL;// список пуст**

**sl[0]='\0';**

**j=0;**

**for(i=0;i<=strlen(s);i++)**

**if((s[i]==' ')||(s[i]=='\0'))**

**{ sl[j]='\0';**

**m=atoi(sl); // преобразование слова в число**

**j=0;**

**sl[0]='\0';**

**if (first==NULL) // если список пуст**

**{ first=new elem;**

**first->num=m;**

**first->next=NULL;**

**last=first;**

**}**

**else // иначе**

**{ n=new elem;**

**n->num=m;**

**if(n->num<first->num) // если новое число меньше**

**// первого элемента**

**{n->next=first;**

**first=n;**

**}**

**else // Иначе ищем место вставки**

**{ fl=0; // флаг того, что найдено место вставки**

**p=first;**

**q=first;**

**while((q=q->next)!=NULL&&!fl)**

**{**

**if(n->num<q->num)**

**{n->next=p->next;**

**p->next=n;**

**fl=1; // элемент поставлен на место**

**}**

**else p=q;**

**}**

**if(!fl) // если место не найдено, добавляем**

**// элемент в конец списка**

**{n->next=NULL;**

**p->next=n;**

**last=n;**

**}**

**}**

**}**

**}**

**else**

**{ sl[j]=s[i];**

**j++;**

**}**

**printf("Spisok\n");**

**q=first;**

**while(q!=NULL)**

**{printf("%6d",q->num);**

**q=q->next;**

**}**

**printf("\n");**

**q = first;**

**puts("Delete spisok");**

**while (q!=NULL)**

**{**

**n = q;**

**q = q->next;**

**delete n;**

**}**

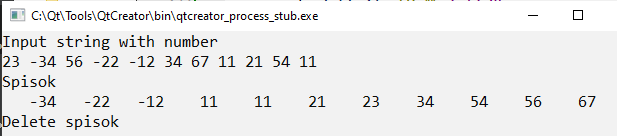
**first=n=p=last=NULL;**

**getch();**

**return 0;**

**}**

**Пимер работы программы**

****

**sortvvod**

**ПРИМЕР 3**

**С клавиатуры вводятся целые числа. Из чисел построить список и вывести его на экран. Упорядочить список по возрастанию его элементов. Вывести новыйсписок.**

Для ввода целых чисел воспользуемся строковым типом данных и функцией преобразования строку в целое число.

Печать списка и сортировку выделим в подпрограммы процедуры. Формирование и удаление списка выполним в основной программе.

В подпрограмме сортировки используем метод обменов (пузырьковая сортировка). Для реализации выберем однонаправленный список-стек.

**#include<stdio.h>**

**#include<stdlib.h>**

**#include<string.h>**

**#include<conio.h>**

**// описание элемента списка**

**struct item {**

**item \*next;**

**int data;**

**};**

**void print\_list(const item \*first)**

**{**

**for (const item \*cur = first; cur; cur = cur->next) {**

**printf("%d ", cur->data);**

**}**

**puts("");**

**}**

**void sort\_list(item \*\*first){**

**item \*cur, \*prev, \*next;**

**bool flag;**

**if (\*first)**

**{**

**flag = true;**

**cur = \*first;**

**while (flag)**

**{**

**flag = false;**

**prev = NULL;**

**cur = \*first;**

**next = cur->next;**

**while (next)**

**{ if (next->data < cur->data)**

**{**

**flag = true;**

**cur->next = next->next;**

**next->next = cur;**

**cur = next;**

**if (prev) prev->next = cur;**

**else \*first = cur;**

**}**

**prev = cur;**

**cur = cur->next;**

**next = cur->next;**

**}**

**}**

**}**

**}**

**int main()**

**{ char str[0xF];// строка для ввода чисел**

**item \*first, \*cur, \*tmp;**

**first = NULL;**

**puts("Input integer value or ENTER");**

**while (gets(*str*), strlen(str)!=0) {**

**cur = new item;**

**cur->data = atoi(str);**

**cur->next = first;**

**first = cur;**

**}**

**puts("\nInputed spisok integer value");**

**print\_list(first);**

**puts("\nSorted spisok integer value"); sort\_list(*&first*);**

**print\_list(first);**

**cur = first;**

**while (cur)**

**{**

**tmp = cur;**

**cur = cur->next;**

**delete tmp;**

**}**

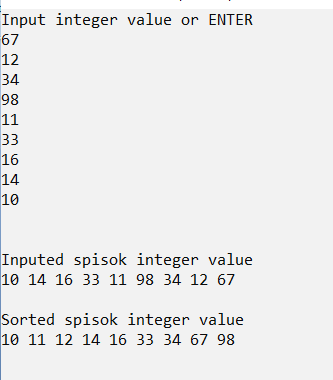
**getch();**

**first=tmp=NULL;**

**return 0;**

**}**

**Пример работы программы**

****

**SortList**

**Пример 4**

**С клавиатуры вводится список сотрудников.**

**Построить из вводимых фамилий список, отсортированный по алфавиту.**

Для сортировки списка воспользуемся сортировкой при вводе (метод вставки) Для реализации выберем однонаправленный список.

**#include<stdio.h>**

**#include<string.h>**

**#include <conio.h>**

**struct rec{**

**char fam[22];**

**rec \*p;};**

**int main()**

**{rec \*n,\*first,\*q,\*p,a;**

**int flag;**

**first=new rec;**

**first->p=NULL;**

**puts("Input string or empty string");**

**gets(*first->fam*); // ввод первой фамилии**

**gets(*a.fam*);**

**while(a.fam[0]!='\0') // пока есть фамилии**

**{**

**n=new rec;**

**strcpy(*n->fam*,a.fam);**

**if(strcmp(n->fam,first->fam)<0) // если фамилия**

**// меньше первого элемента**

**{n->p=first;**

**first=n;**

**}**

**else // Иначе ищем место вставки фамилии**

**{**

**flag=0; // флаг для поиска**

**p=first;**

**q=first;**

**while((q)!=NULL&&!flag)**

**{if(strcmp(n->fam,q->fam)<0)**

**{n->p=p->p;**

**p->p=n;**

**flag=1;} // место вставки найдено**

**else {p=q;q=q->p;}**

**}**

**if(!flag)// Место не найдено – вставляем в конец списка**

**{n->p=NULL;**

**p->p=n;**

**}**

**}**

**gets(*a.fam*); // ввод новой фамилии**

**}**

**puts("Sorted spisok");**

**q=first;**

**while(q!=NULL)**

**{**

**puts(q->fam);**

**q=q->p;**

**}**

**q = first;**

**puts("Delete spisok");**

**while (q!=NULL)**

**{**

**p = q;**

**q = q->p;**

**delete p;**

**}**

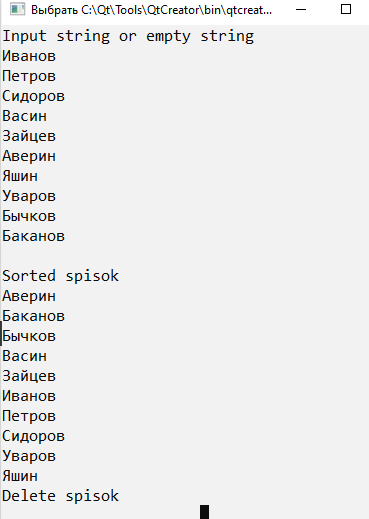
**first=p=n=NULL;**

**getch();**

**return 0;**

**}**

**Пример работы программы**

****