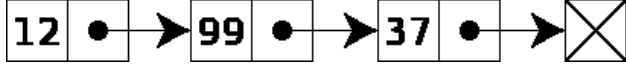


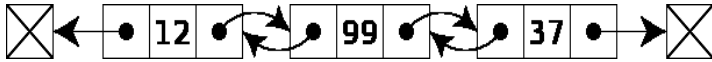
Проблеми при въвеждане и изтриване елементи в масив
Свързани структури – елементите не са в последователни адреси на паметта – информация за местоположението на елементите

Свързан списък(linked list): структура от последователност от възли (nodes) съдържащи връзка(и)(reference, link) към следващите възли.

Едносвързан списък - всеки възел - две полета(данни и връзка):



Двойно свързан списък- всеки възел включва данни и връзки към левия и десния възел



Базова структура – използва се за създаване други като например стекове и опашки.

Предимство – лесно добавяне и изтриване на елементи.

Недостатък – няма пряк достъп до елементите.

Пример : Четене в подреден едносвързан списък

1. Създаване на бинарен файл от структури – въвеждане от клавиатурата формиране на структура и запис във файл до въвеждане на „*” като име. ([свали от тук](#))

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct {
    char name [30];
    int n;
}Person;
int main(){
    FILE *f;
    Person p;
    if((f=fopen("D:\\Work\\person.dat","wb"))!=NULL){
        do{
            printf ("next name:");
            scanf("%29s",p.name);
            if(!strcmp(p.name,"*"))break;
            printf("next number:");
            scanf("%d",&(p.n)); /* control of the input! */
            fwrite(&p, sizeof(Person),1,f);
            printf("Writing the structure in the file\n");
        }while (1);
        fclose(f);
        printf("The file is created");
    }
    return 0;
}
```

/*Четене от файла и въвеждане в подреден свързан списък */

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct {
    char name [30];
    int n;
}Person;
```

```
typedef struct node { /*едносвързан */
    Person prs;
    struct node *next;
}node;

node * makeSrtLst(node * head, FILE *f){
    node * p,*crnt,*prev;
    do{
        p=(node *)malloc(sizeof(node)); /* if p==0 */
        p->next= NULL;
        if(!fread(&(p->prs),sizeof(Person),1,f)){
            free(p);
            break;
        }
        /* find the correct place – sorted by n */
        for(prev=NULL,crnt=head;
            crnt && (crnt->prs.n > p->prs.n);
            prev=crnt, crnt=crnt->next);
        // put in the list
        if(prev)prev->next=p;
        else head = p;
        p->next=crnt;

        /* or put directly in the head of the list – no sort
        p->next=head;
        head = p;
        */

    }while(1);
    return head;
}

void prt(node *crnt){
    printf("the list is:\n");
    while(crnt){
        printf("%s has %d\n",crnt->prs.name,crnt->prs.n);
        crnt=crnt->next;
    }
}

node * free_m(node *crnt){
    node* next;
    printf("\n memory free\n");
    while(crnt){
        next = crnt->next; free(crnt); crnt=next;
    }
    return NULL;
}

int main(){
    FILE *f;
    node *head=NULL;
    if (!f=fopen("D:\\Work\\person.dat","rb"))){
        return 1;
    }
    head=makeSrtLst(head,f);
    prt(head);
    head= free_m(head);
    return 0;
}
(свали файла от тук)
```