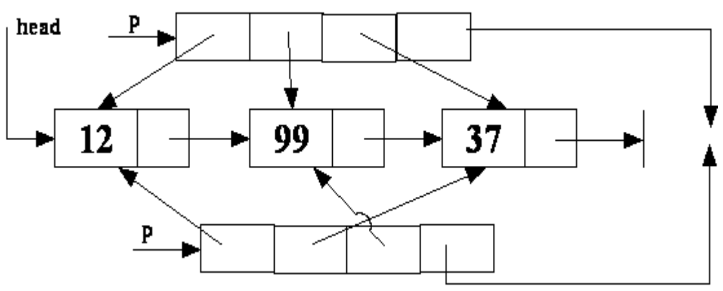
Сортиране на свързан списък (с масив от указатели)



1.Създаване на бинарен файл от структури – въвеждане от клавиатурата формиране на структура и запис във файл до въвеждане на „*” като име. (свали файла 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;
}
```

1. Четене от файла и въвеждане в свързан списък (свали тук)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct {
  char name [30];
  int n;
}Person;
typedef struct node {
   Person prs;
   struct node *next;
}node;
node * makeLst(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;
    }
/*            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;
  }
}
void srtNm(node **ph){
  int ok;  node **p; node *hlp;
  do{
     ok=1;
     for(p=ph;*(p+1);p++){
          if(((*p)->prs.n) > ((*(p+1))->prs.n)){
            hlp=*p; *p=*(p+1); *(p+1)=hlp; ok=0;
          }
     }
  }while(!ok);
}
void prtSrt(node *head){
   node **ph= NULL,**p;   // dynamic array of pointers to node
   node *crnt;  int number=0;
   for(crnt=head;crnt; crnt=crnt->next)number++; //how many nodes
   number++;  // for the end NULL
   ph=(node **)malloc(number*sizeof(node*));
   if(!ph) {/*…*/}
   for(p=ph,crnt=head;crnt; crnt=crnt->next) *p++=crnt;
   *p=NULL;
   srtNm(ph);
   for(p=ph; *p;p++){
     printf("%s  has %d\n",(*p)->prs.name,(*p)->prs.n);
   }
   free(ph);
}
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=makeLst(head,f);
   prt(head);
   printf("\nSorted\n");
   prtSrt(head);
   head= free_m(head);
   return 0;
}
```

Съз даване на повече от един масив от указатели за сортиране:
(свали програмата от тук)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct {
   char name [30];
   int n;
}Person;
typedef struct node {
   Person prs;
   struct node *next;
}node;
node * makeLst(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;
      }
      /* 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;
   }
}
void srtNm(node **ph){
   int ok; node **p; node *hlp;
   do{
      ok=1;
      for(p=ph;*(p+1);p++){
         if(((*p)->prs.n) > ((*(p+1))->prs.n)){
            hlp=*p; *p=*(p+1); *(p+1)=hlp; ok=0;
         }
      }
   }while(!ok);
}

node** make_ar(node *head, node **ps){
   node *crnt; int number=0; node**p;
   for(crnt=head;crnt; crnt=crnt->next)number++; //how many
   number++; // for the end NULL
   ps=(node **)malloc(number*sizeof(node*));
   if(!ps) {/*…*/}
   for(p=ps,crnt=head;crnt; crnt=crnt->next) *p++=crnt;
   *p=NULL;
   return ps;
}
```

```
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, **p;
   node **ps1=NULL,**ps2=NULL; //arays of pointers
   if (!(f=fopen("person.dat","rb"))){
      puts("file not found"); return 1;
   }
   head=makeLst(head,f);
   prt(head);
   printf("\nSorted\n");
   ps1 = make_ar(head,ps1);
   srtNm(ps1);
   for(p=ps1; *p;p++){
      printf("%s has %d\n",(*p)->prs.name,(*p)->prs.n);
   }
   free(ps1);
   head= free_m(head);
   return 0;
}
```