```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define NBEMP 100
typedef struct{
    int num;
    char pnom[20];
    char nom[20];
    float salr;
    char grad[3];
    char sup[3];
}EMP;
EMP* Liref(FILE* ptr);
void Ecrire(FILE* ptr,EMP* T);
void tri_bulle(EMP* T,int mode_tri, int mode_algo);
void tri_selection(EMP* T,int mode_tri, int mode_algo);
void tri_insertion(EMP* T,int mode_tri, int mode_algo);
void swap(EMP* T,int i,int j);/*permer de swaper les valeurs de 2 élément d'un tableau T*/
int issup(char* S1,char* S2);
int condition(EMP* T,int i,int j,int mode_tri,int choix_tri);
/*permer de decider le mode de trie selon les arguments:
mod_tri(croissant/décroissant)
choix_tri(selon le prénom,nom,...)*/
int condition_inser(EMP* T,int i,EMP val,int mode_tri,int choix_tri);
/*elle fontion comme 'contiotion()' mais cette fonction est pour
l'algorithme de tri par insertion */
int main()
{
    int rep1,rep2,rep3,i;
    EMP* S;
    FILE* fp;
    fp=fopen("liste_employé[1].txt","rt");
    if(fp==NULL){
        printf("ERREUR d'ouvrer le fichier");
        exit(33);
    }
    S=Liref(fp);
    fclose(fp);
    for(i=0;i<NBEMP;i++)
        printf("%d:%s:%s:%f:%s:%s\n",S[i].num,S[i].pnom,S[i].nom,S[i].salr,S[i].grad,S[i].sup);
    printf("\nEntrer le nombre convenable pour afficher la liste selon:\n");
    printf("\t1:Le Numero\n\t2:Le Prenom\n\t3:Le Nom\n\t4:Le salaire/grad/emploiyee superieur\n");
    do{scanf("%d",&rep1);}while(rep1<1&&rep1>4);
    printf("\nPar order:\n\t1:croissant\n\t2:decroissant\n");
    do{scanf("%d",&rep2);}while(rep2!=1&&rep2!=2);
    printf("Quelles algorithme voulez vous utiliser?\n\t1:Le tri bulle\n\t2:Le tri selection\n\t3:le tri insertion\n");
    do{scanf("%d",&rep3);}while(rep3<1&&rep3>3);
    system("cls");
    switch(rep3){
        case 1: tri_bulle(S,rep2,rep1);
                break;
        case 2: tri_selection(S,rep2,rep1);
                break;
        case 3: tri_insertion(S,rep2,rep1);
                break;
    }
    fp=fopen("list triee.txt","wt");
    Ecrire(fp,S);
    fclose(fp);
    for(i=0;i<100;i++)
        printf("%d:%s:%s:%f:%s:%s\n",S[i].num,S[i].pnom,S[i].nom,S[i].salr,S[i].grad,S[i].sup);
    return 0;
}
EMP* Liref(FILE* ptr)
{
    EMP* T;
    char tmp[100],c;
    int i,j,k,nbr;
```

```c
    i=j=k=nbr=0;
    T=(EMP*)malloc(100*sizeof(EMP));
    while((c=getc(ptr))!=EOF){
        if(c!=':'&&c!='\n'){
            tmp[k++]=c;
            nbr++;
        }else if(c==':'){
            tmp[k]='\0';
            switch(j){
                case 0: T[i].num=atoi(tmp); /*atoi(Ascii to intger) fontion permer de
                tronsformer
                                            un nombre string à un intger*/
                        break;
                case 1: strcpy(T[i].pnom,tmp);
                        break;
                case 2: strcpy(T[i].nom,tmp);
                        break;
                case 3: T[i].salr=atof(tmp);/*atoi(Ascii to float) fontion permer de
                tronsformer
                                            un nombre string à un float*/
                        break;
                case 4: strcpy(T[i].grad,tmp);
                        break;
            }
           k=nbr=0;
           j++;

        }else if(c=='\n'){
        tmp[k]='\0';
        strcpy(T[i].sup,tmp);
        j=k=nbr=0;
        i++;
        }
    }
    tmp[k]='\0';
        strcpy(T[i].sup,tmp);
        return T;
}
void Ecrire(FILE* ptr,EMP* T)
{
    int i;
    for(i=0;i<NBEMP;i++)
        fprintf(ptr,"%d:%s:%s:%f:%s:%s\n",T[i].num,T[i].pnom,T[i].nom,T[i].salr,T[i].
        grad,T[i].sup);
}
void tri_bulle(EMP* T,int mode_tri, int choix_tri)
{
    int i,j,flag;
    for(i=0;i<NBEMP;i++){
            flag=0;
        for(j=0;j<NBEMP-i-1;j++){
            if(condition(T,j,j+1,mode_tri,choix_tri)){
                flag=1;
                swap(T,j,j+1);
            }
        }
        if(!flag) break;
    }
}
void tri_selection(EMP* T,int mode_tri, int choix_tri)
{
    int i,j,imax;
    for(i=0;i<NBEMP-1;i++){
        imax=i;
        for(j=i+1;j<NBEMP;j++){
            if(condition(T,j,imax,mode_tri,choix_tri))
                swap(T,j,imax);
        }
    }
}
void tri_insertion(EMP* T,int mode_tri, int choix_tri)
{
    int i,ind,tmpi;
```

```
    EMP tmp;
    for(i=1;i<NBEMP;i++){
        tmp.salr=T[i].salr;
        tmp.num=T[i].num;
        sprintf(tmp.pnom,"%s",T[i].pnom);
        sprintf(tmp.nom,"%s",T[i].nom);
        sprintf(tmp.grad,"%s",T[i].grad);
        sprintf(tmp.sup,"%s",T[i].sup);
        ind=i;
        while(ind>0&&condition_inser(T,ind-1,tmp,mode_tri,choix_tri)){
            T[ind].salr=T[ind-1].salr;
            T[ind].num=T[ind-1].num;
            sprintf(T[ind].pnom,"%s",T[ind-1].pnom);
            sprintf(T[ind].nom,"%s",T[ind-1].nom);
            sprintf(T[ind].grad,"%s",T[ind-1].grad);
            sprintf(T[ind].sup,"%s",T[ind-1].sup);
            ind--;
        }
        T[ind].salr=tmp.salr;
        T[ind].num=tmp.num;
        sprintf(T[ind].pnom,"%s",tmp.pnom);
        sprintf(T[ind].nom,"%s",tmp.nom);
        sprintf(T[ind].grad,"%s",tmp.grad);
        sprintf(T[ind].sup,"%s",tmp.sup);
    }
}
void swap(EMP* T,int i,int j)
{
    int tmpi;
    float tmpf;
    char tmps[50];

    tmpi=T[i].num;
    T[i].num=T[j].num;
    T[j].num=tmpi;
    sprintf(tmps,"%s",T[i].pnom);
    sprintf(T[i].pnom,"%s",T[j].pnom);
    sprintf(T[j].pnom,"%s",tmps);
    sprintf(tmps,"%s",T[i].nom);
    sprintf(T[i].nom,"%s",T[j].nom);
    sprintf(T[j].nom,"%s",tmps);
    tmpf=T[i].salr;
    T[i].salr=T[j].salr;
    T[j].salr=tmpf;
    sprintf(tmps,"%s",T[i].grad);
    sprintf(T[i].grad,"%s",T[j].grad);
    sprintf(T[j].grad,"%s",tmps);
    sprintf(tmps,"%s",T[i].sup);
    sprintf(T[i].sup,"%s",T[j].sup);
    sprintf(T[j].sup,"%s",tmps);
}
int issup(char* S1,char* S2)
{
    int i,j,l1,l2;
    l1=strlen(S1);
    l2=strlen(S2);
    i=j=0;
    while(i<l1&&j<l2){
        if(S1[i]==' '&&S1[i]!=S2[j]){
            i++;
            continue;
        }else if(S2[j]==' '&&S1[i]!=S2[j]){
            j++;
            continue;
        }
        if(S1[i]!=S2[j])
            break;
        i++;
        j++;
    }
    if(i<l1&&j<l2)
        if(S1[i]<S2[j])
            return 0;
```

```
212             else return 1;
213         if (i<l1) return 1;
214         else return 0;
215     }
216     int condition(EMP* T,int i,int j,int mode_tri, int choix_tri)
217     {
218         switch (choix_tri){
219         case 1: if(mode_tri==1) return (T[i].num>T[j].num);
220                 return (T[i].num<T[j].num);
221         case 2: if(mode_tri==1) return (!issup(T[i].pnom,T[j].pnom));
222                 return (issup(T[i].pnom,T[j].pnom));
223         case 3: if(mode_tri==1) return (!issup(T[i].nom,T[j].nom));
224                 return (issup(T[i].nom,T[j].nom));
225         case 4: if(mode_tri==1) return (T[i].salr>T[j].salr);
226                 return (T[i].salr<T[j].salr);
227         }
228     }
229     int condition_inser(EMP* T,int i,EMP val,int mode_tri,int choix_tri)
230     {
231         switch (choix_tri){
232         case 1: if(mode_tri==1) return (T[i].num>val.num);
233                 return (T[i].num<val.num);
234         case 2: if(mode_tri==1) return (!issup(T[i].pnom,val.pnom));
235                 return (issup(T[i].pnom,val.pnom));
236         case 3: if(mode_tri==1) return (!issup(T[i].nom,val.nom));
237                 return (issup(T[i].nom,val.nom));
238         case 4: if(mode_tri==1) return (T[i].salr>val.salr);
239                 return (T[i].salr<val.salr);
240         }
241     }
242
```