Laboratorul 10 – Prelucrarea fișierelor prin funcțiile de nivel înalt

```
/// 3.2. Se citeşte de la tastatură un text care se scrie într-un fişier "text.dat". Să
se afişeze apoi conţinutul fişierului (fără a cunoaşte lungimea maximă a liniilor),
fiecare linie fiind precedată de numărul de ordine al ei.

#include <stdio.h>
#include <stdlib.h>

void creeaza(const char *nume){
    char s[1000];
    FILE* pf = fopen(nume, "w");

    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    printf("Creare %s: Dati un text, iar la sfarsit Ctrl + Z: \n", nume);

    while(scanf("%s", &s)==1) fprintf(pf, "%s\n", s);

    fclose(pf);
}

void afiseaza(const char *nume){
    int i=0;
    char s[50];
    FILE* pf = fopen(nume, "r");

    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    while(fgets(s, 50, pf)!=NULL) printf("%d\t%s", ++i, s);

    fclose(pf);
}

int main()
{

    creeaza("text.dat");
    afiseaza("text.dat");

    return 0;
}
```

```c
/// 3.3. De la tastatură se citesc partea reală și partea imaginară pentru n numere complexe. Să se creeze un fișier care conține numerele complexe citite, fiecare număr având partea reală, partea imaginară, modulul și argumentul său. Se va afișa conținutul fișierului.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

typedef struct { float re, im; double mod, arg; } Complex;

void creeaza(const char *nume){
    int n, i;
    Complex z;
    FILE* pf = fopen(nume, "w");

    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    printf("Numarul de numere complexe: ");
    scanf("%d", &n);

    for(i=0; i<n; i++){
        printf("[%d].re = ", i);
        scanf("%f", &z.re);

        printf("[%d].im = ", i);
        scanf("%f", &z.im);

        z.mod = sqrt(z.re*z.re + z.im*z.im);
        z.arg = atan2(z.im, z.re);

        fprintf(pf, "%f %f %f %f\n", z.re, z.im, z.mod, z.arg);
    }

    fclose(pf);
}

void afiseaza(const char *nume){
    Complex z;
    FILE *pf = fopen(nume, "r");

    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    while(fscanf(pf, "%f%f%lf%lf", &z.re, &z.im, &z.mod, &z.arg)==4){
        printf("%.2f + i*%.2f\t Modul: %.3f", z.re, z.im, z.mod);
        printf("\tArg: %.2f grade\n", z.arg*180/M_PI);
    }

    fclose(pf);
}

int main()
```

```
{
    char nume[]="fis.txt";

    creeaza(nume);
    afiseaza(nume);

    return 0;
}
```

```
/// 3.4. Să se scrie programul pentru concatenarea a două sau mai multe fişiere ce conţin numere reale. Se va tipări informaţia din fişierul rezultat.

#include <stdio.h>
#include <stdlib.h>

void creeaza(const char *nume){
    float x;
    FILE* pf = fopen(nume, "w");

    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    printf("Creare %s: Dati numere reale, iar la sfarsit Ctrl + Z: \n", nume);

    while(scanf("%f", &x)==1) fprintf(pf, "%f\n", x);

    fclose(pf);
}

void afiseaza(const char *nume){
    char s[50];

    FILE* pf = fopen(nume, "r");
    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    while(fgets(s, 50, pf)!=NULL) printf("%s", s);

    fclose(pf);
}

void concat(FILE* pr, const char *nume){
    float x;
    FILE* pf = fopen(nume, "r");

    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    while(fscanf(pf, "%f", &x)==1) fprintf(pr, "%f ", x);

    fprintf(pr, "\n");
```

```
    fclose(pf);
}

int main()
{
    int n, i;
    char nume[20], rez[]="rezultat.txt";

    FILE* pr=fopen(rez, "w");

    if(pr==NULL){
        perror(rez);
        exit(1);
    }

    printf("Numar fisiere: ");
    scanf("%d", &n);

    for(i=1; i<=n; i++){
        sprintf(nume, "fis%d.txt", i);
        creeaza(nume);
        concat(pr, nume);
    }

    fclose(pr);
    afiseaza(rez);

    return 0;
}
```

```
/// 3.5. Să se creeze un fişier care să conţină produsele unui magazin. Un produs este
reprezentat printr-o structură ce conţine codul produsului, denumirea, unitatea de
măsură, cantitatea, preţul unitar. Scrieţi funcţii de intrare şi de ieşire a produselor
magazinului. Plecând de la acest fişier, să se obţină un fişier sortat după cod.

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 100

typedef struct { char cod[N], denumire[N], unitate[N]; float pret, cant; } Produs;

void *creeaza(const char *nume, int *n){
    int i, l;

    FILE* pf = fopen(nume, "w");
    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    printf("Introduceti numarul de produse: ");
    scanf("%d%*c", n);

    Produs *p=malloc(sizeof(Produs)*(*n)+1);

    for(i=0; i<*n; i++){
        printf("\nProdusul #%d: \n", i);

        printf("\tCodul produsului #%d: ", i);
        fgets(p[i].cod, N, stdin);

        l=strlen(p[i].cod);
        if(p[i].cod[l-1]=='\n') p[i].cod[l-1]='\0';

        printf("\tDenumirea produsului #%d: ", i);
        fgets(p[i].denumire, N, stdin);

        l=strlen(p[i].denumire);
        if(p[i].denumire[l-1]=='\n') p[i].denumire[l-1]='\0';

        printf("\tUnitarea de masura #%d: ", i);
        fgets(p[i].unitate, N, stdin);

        l=strlen(p[i].unitate);
        if(p[i].unitate[l-1]=='\n') p[i].unitate[l-1]='\0';

        printf("\tCantitatea #%d: ", i);
        scanf("%f%*c", &p[i].cant);

        printf("\tPretul #%d: ", i);
        scanf("%f%*c", &p[i].pret);

        fprintf(pf, "%s\t%s\t%s\t%.2f\t%.2f\n", p[i].cod, p[i].denumire, p[i].unitate,
p[i].cant, p[i].pret);
    }

    fclose(pf);
```

```
    return p;
}

void sortare(Produs *p, int n, const char *lista){
    int i, sortat=0, k=0;
    k=n;
    Produs aux;

    FILE *pf = fopen(lista, "w+");

    if(pf==NULL){
        perror(pf);
        exit(1);
    }

    while(!sortat){
        sortat=1;
        for(i=1; i<n; i++)
            if(strcmp(p[i-1].cod, p[i].cod)>=0){
                aux=p[i-1];
                p[i-1]=p[i];
                p[i]=aux;
                sortat=0;
            }
        n--;
    }

    fprintf(pf, "-----PRODUSE-----\n");

    for(i=0; i<k; i++) fprintf(pf, "%s\t%s\t%s\t%.2f\t%.2f\n", p[i].cod, p[i].denumire,
p[i].unitate, p[i].cant, p[i].pret);

}

int main()
{
    Produs *p;
    int n;

    p=creeaza("date.txt", &n);
    sortare(p, n, "produse.txt");

    return 0;
}
```

```c
/// 3.6. Să se scrie un program pentru admiterea la facultate în anul I. Programul va cuprinde crearea fişierului cu candidaţii înscrişi. În final trebuie să se obţină fişierele cu candidaţii admişi şi cei respinşi, pe baza numărului de locuri disponibile şi a mediei obţinute ((bacalaureat + 2 * test_matem) / 3.0).

#include <stdio.h>
#include <stdlib.h>
#define N 60

typedef struct { char nume[N]; float bac, test, medie; } Candidat;

void *creeaza(const char *nume, int *n, int *disp){
    int i, l;

    FILE* pf = fopen(nume, "w");
    if(pf==NULL){
        perror(nume);
        exit(1);
    }

    printf("Numarul canditatilor: ");
    scanf("%d%*c", n);

    Candidat *c=malloc(sizeof(Candidat)*(*n)+1);

    printf("Locuri disponibile: ");
    scanf("%d%*c", disp);

    fprintf(pf, "%d %d\n", *n, *disp);

    for(i=0; i<*n; i++){
        printf("\tNumele #%d: ", i);
        fgets(c[i].nume, N, stdin);

        l=strlen(c[i].nume);
        if(c[i].nume[l-1]=='\n') c[i].nume[l-1]='\0';

        printf("\tMedia bac #%d: ", i);
        scanf("%f%*c", &c[i].bac);

        printf("\tNota test mate #%d: ", i);
        scanf("%f%*c", &c[i].test);
        printf("\n");

        c[i].medie=(c[i].bac+2*c[i].test)/3;

        fprintf(pf, "%s %.2f\n", c[i].nume, c[i].medie);
    }

    fclose(pf);

    return c;
}

void sortare(Candidat *c, int n, int disp, const char *fisierAdmisi, const char
*fisierRespinsi){
    int i, sortat=0, k=n;
    Candidat aux;
```

```
    FILE *pfAdmisi = fopen(fisierAdmisi, "w");
    FILE *pfRespinsi = fopen(fisierRespinsi, "w");

    if(pfAdmisi==NULL){
        perror(pfAdmisi);
        exit(1);
    }

    if(pfRespinsi==NULL){
        perror(pfRespinsi);
        exit(1);
    }

    while(!sortat){
        sortat=1;
        for(i=1; i<n; i++)
            if(c[i-1].medie<c[i].medie){
                aux=c[i-1];
                c[i-1]=c[i];
                c[i]=aux;
                sortat=0;
            }
        n--;
    }

    fprintf(pfAdmisi, "-----CANDIDATI ADMISI-----\n");
    fprintf(pfRespinsi, "-----CANDIDATI RESPINSI-----\n");

    for(i=0; i<disp; i++) fprintf(pfAdmisi, "%d\t%s\t%.2f \n", i+1, c[i].nume,
c[i].medie);
    for(i=disp; i<=k; i++) fprintf(pfRespinsi, "%d\t%s\t%.2f \n", i-disp+1, c[i].nume,
c[i].medie);
}


int main()
{
    Candidat *c;
    int n, disp;

    c=creeaza("date.txt", &n, &disp);
    sortare(c, n, disp, "admisi.txt", "respinsi.txt");

    return 0;
}
```