

Na **Desktop**-u u direktorijumu **Rad** kreirati direktorijum **ImePrezime_BrIndeksa** i unutar njega sačuvati programe koji sadrže rešenja datih zadataka. Rešenje 1. zadatka mora da se nalazi u fajlu **Zadatak1.c**, rešenje 2. zadatka mora da se nalazi u fajlu **Zadatak2.c**, rešenje 3. zadatka mora da se nalazi u fajlu **Zadatak3.c**. Od tri ponudjena zadatka birate dva koja ćete raditi.

Napisati C program koji:

1. kao parametar pri startovanju program dobija imena ulazne i izlazne datoteke. U ulaznoj datoteci se u prve dve linije nalaze celi brojevi m i n , a zatim do kraja datoteke celi brojevi manji od 1000. U izlaznu datoteku prepisati brojeve iz ulazne datoteke, ali tako da formiraju matricu sa n kolona, pri čemu prva vrsta sarži negativne brojeve iz ulazne datoteke, druga pozitivne i sve tako naizmenično dok se ne popune sve vrste. Svaki broj se zapisuje na 4 pozicije. Ukoliko u datoteci nema dovoljno negativnih brojeva njih zamenjuju nule, a ukoliko nema dovoljno pozitivnih njih zamenjuju jedinice.

```
#include<stdio.h>
main(int argc,char*argv[]){
FILE *fin,*fout;
int poz[100],neg[100],np=0,nn=0,n,m,ip,in,i,j,k;
fin=fopen(argv[1],"r");
if(fin==NULL) exit(1);
fscanf(fin,"%d",&m,&n);
for(i=0;i<m*n;i++){
fscanf(fin,"%d",&k);
if(k>0) poz[np++]=k;
else neg[nn++]=k;
}
fclose(fin);
fout=fopen(argv[2],"w");
if(fout==NULL) exit(1);
ip=0; in=0;
while(ip<np || in<nn){
for(i=0;i<n;i++){
if(in<nn) fprintf(fout,"%4d",neg[in++]);
else fprintf(fout,"%4d",0);
fprintf(fout,"\n");
if(ip<np){
for(i=0;i<n;i++){
if(ip<np) fprintf(fout,"%4d",poz[ip++]);
else fprintf(fout,"%4d",1);
fprintf(fout,"\n");
}
}
}
fclose(fout);
}
```

2. kao parametar pri startovanju programa dobija ime tekstualne datoteke sa podacima o prozorima i vratima (oznaka "PROZOR" ili "VRATA", širina i visina – realni brojevi, cm) i to po jedan podatak u jednom redu. Definirati strukturu **prozor_vrata** koja sadrži sledeće podatke navedene podatke. Vrata se prave od dasaka fiksne širine 30cm i dovoljne dužine, nadovezivanjem dasaka. Ostatak daske koji se dobije pri pravljenju vrata ne može da se upotrebi za druga vrata. Program sadrži funkciju **Vrata** kojom se određuje broj dasaka potreban za pravljenje svih vrata. Program sadrži i funkciju **Prozor** koja određuje najmanju površinu prozora. U glavnom programu učitati podatke i ispisati vrednosti dobijene iz fukcija **Prozor** i **Vrata**.

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

struct prozor_vrata{
char tip[6];
float sirina,visina;};
```

```

int Vrata(struct prozor_vrata vp[],int n){
    int i,d=0;
    float k;
    for(i=0;i<n;i++)
        if(!strcmp(vp[i].tip,"VRATA")) {
            k=vp[i].sirina/30;
            if(k==(int)k) d=d+(int)k;
            else d=d+(int)k+1;
        }
    return d;
}

float Prozor(struct prozor_vrata vp[],int n){
    int i=0;
    float min=0;
    while(i<n && strcmp(vp[i].tip,"PROZOR")) i++;
    if(i<n) {
        min=vp[i].sirina*vp[i].visina;
        for(;i<n;i++)
            if(vp[i].sirina*vp[i].visina<min) min=vp[i].sirina*vp[i].visina;
    }
    return min;
}

main(int argc,char*argv[]){
    FILE *fin;
    struct prozor_vrata vp[100];
    int i,n=0;
    char s[6];
    float s1,v1;

    fin=fopen(argv[1],"r");
    if(fin==NULL) exit(1);

    while(!feof(fin)){
        if(fscanf(fin,"%s%f%f",s,&s1,&v1)==3) {
            strcpy(vp[n].tip,s);
            vp[n].sirina=s1;
            vp[n++].visina=v1;
        }
        else break;
    }
    fclose(fin);
    printf("%d\n",Vrata(vp,n));
    printf("%.2f\n",Prozor(vp,n));
}

```

3. učitava niz realnih brojeva od **n** elemenata. Napisati rekurzivnu funkciju **ZbirOd** koja transformiše učitani niz tako što svaki element niza zamenjuje zbirom elemenata niza od posmatranog elementa do kraja (uključujući i posmatrani element). U glavnom delu programa ispisati dobijeni niz.

```

#include<stdio.h>

float ZbirOd(float a[],int i,int n){
    if(i==n-1) return a[i];
    else a[i]=a[i]+ZbirOd(a,i+1,n);
}

```

```

    }

    main(){
        int i,n;
        float a[50];
        scanf("%d",&n);
        for(i=0;i<n;i++) scanf("%f",&a[i]);
        ZbirOd(a,0,n);
        for(i=0;i<n;i++) printf("%.2f\n",a[i]);
    }

```

Primer: Ulaz: N = 5 niz: 1 2 3 4 5

Izlaz: 15 14 11 9 5

Broj poena: 1. zadatak – 7 poena, 2. zadatak – 13 poena, 3. zadatak – 12 poena

Vreme izrade: 60 minuta

Strukture podataka i algoritmi 1 - II kolokvijum

04.06.2012

Na **Desktop**-u u direktorijumu **Rad** kreirati direktorijum **ImePrezime_BrIndeksa** i unutar njega sačuvati programe koji sadrže rešenja datih zadataka. Rešenje 1. zadatka mora da se nalazi u fajlu **Zadatak1.c**, rešenje 2. zadatka mora da se nalazi u fajlu **Zadatak2.c**, rešenje 3. zadatka mora da se nalazi u fajlu **Zadatak3.c**. Od tri ponudjena zadatka birate dva koja ćete raditi.

Napisati C program koji:

1. kao parametar pri startovanju program dobija imena ulazne i izlazne datoteke. U ulaznoj datoteci se u prve dve linije nalaze celi brojevi m i n, a zatim do kraja datoteke celi brojevi manji od 1000. U izlaznu datoteku prepisati brojeve iz ulazne datoteke, ali tako da formiraju matricu sa n kolona, pri čemu svaka vrsta počinje pozitivnim brojem a zatim idu naizmenično pozitivni i negativni brojevi. Svaki broj se zapisuje na 4 pozicije. Ukoliko u datoteci nema dovoljno negativnih brojeva njih zamenjuju nule, a ukoliko nema dovoljno pozitivnih njih zamenjuju jedinice.

```

#include<stdio.h>
main(int argc,char*argv[]){
    FILE *fin,*fout;
    int poz[100],neg[100],np=0,nn=0,n,m,ip,in,i,j,k;
    fin=fopen(argv[1],"r");
    if(fin==NULL) exit(1);
    fscanf(fin,"%d%d",&m,&n);
    for(i=0;i<m*n;i++){
        fscanf(fin,"%d",&k);
        if(k>0) poz[np++]=k;
        else neg[nn++]=k;
    }
    fclose(fin);
    fout=fopen(argv[2],"w");
    if(fout==NULL) exit(1);
    ip=0; in=0;
    while(ip<np || in<nn){
        for(i=0;i<n;i++) {
            if(i%2==0){
                if(ip<np) fprintf(fout,"%4d",poz[ip++]);
                else fprintf(fout,"%4d",1);
            }
            else {
                if(in<nn) fprintf(fout,"%4d",neg[in++]);
                else fprintf(fout,"%4d",0);
            }
        }
    }
}

```

```

        fprintf(fout, "\n");
    }
    fclose(fout);
}

```

2. kao parametar pri startovanju programa dobija ime tekstualne datoteke sa podacima o prozorima i zavesama (način kačenja zavese – “FALTA” ili “NABOR”, da li zavesa ima draperiju – “IMA” ili “NEMA” i širina prozora – realan broj, cm) i to po jedan podatak u jednom redu. Definirati strukturu **zavesa** koja sadrži navedene podatke. Zavesa koja se kači sa faltom mora da bude oko 3 puta duža nego prozor, a zavesa koja se kači sa naborom mora da bude oko 2,5 puta duža nego prozor. Draperija je iste širine kao zavesa. Prodavac prodaje samo zavese čija je dužina ceo broj metara (minimalna dužina je 1m), pa se širina zavese-draperije zaokružuje na bliži ceo broj. Program sadrži funkciju **Zavese** kojom se određuje ukupna dužina zavesa potrebne za sve prozore. Program sadrži i funkciju **Draperija** koja određuje prosečnu širinu draperija. U glavnom programu učitati podatke i ispisati vrednosti dobijene iz funkcija **Draperija** i **Zavese**.

```

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

struct zavesa{
    char kacenje[6],draperija[5];
    float sirina;
    int sir_zav;};

int Zavese(struct zavesa z[],int n){
    int i,d=0;
    for(i=0;i<n;i++) d+=z[i].sir_zav;
    return d;
}

float Draperija(struct zavesa z[],int n){
    int i,k=0;
    float p=0.0;
    for(i=0;i<n;i++)
        if(!strcmp(z[i].draperija,"IMA")) {
            p+=z[i].sir_zav;
            k++;
        }
    return p/k;
}

main(int argc,char*argv[]){
    FILE *fin;
    struct zavesa z[100];
    int i,n=0;
    char s1[5],s2[4];
    float sir;

    fin=fopen(argv[1],"r");
    if(fin==NULL) exit(1);

    while(!feof(fin)){
        if(fscanf(fin,"%s%s%f",s1,s2,&sir)==3) {
            strcpy(z[n].kacenje,s1);
            strcpy(z[n].draperija,s2);
            z[n++].sirina=sir;
        }
    }
}

```

```

    }
    else break;
}
fclose(fin);
for(i=0;i<n;i++) {
    if(!strcmp(z[i].kacenje,"FALTA"))
z[i].sir_zav=(int)(z[i].sirina*3/100+0.5);
    else z[i].sir_zav=(int)(z[i].sirina*2.5/100+0.5);
    if (z[i].sir_zav==0) z[i].sir_zav=1;
}

printf("%d\n",Zavese(z,n));
printf("%.2f\n",Draperija(z,n));

}

```

3. učitava niz realnih brojeva od **n** elemenata. Napisati rekurzivnu funkciju **PozitivniOd** koja transformiše učitani niz tako što svaki element niza zamenjuje brojem pozitivnih elemenata niza od posmatranog elementa do kraja (uključujući i posmatrani element). U glavnom delu programa ispisati dobijeni niz.

```

#include<stdio.h>

float PozitivniOd(float a[],int i,int n){
    if(i==n) return 0;
    if (a[i]>0) a[i]=1+PozitivniOd(a,i+1,n);
    else a[i]=PozitivniOd(a,i+1,n);
}

main(){
    int i,n;
    float a[50];
    scanf("%d",&n);
    for(i=0;i<n;i++) scanf("%f",&a[i]);
    PozitivniOd(a,0,n);
    for(i=0;i<n;i++) printf("%.2f\n",a[i]);
}

```

Primer: Ulaz: N = 5 niz: 1 -2 3 -4 5 Izlaz: 3 3 2 2 1

Broj poena: 1. zadatak – 7 poena, 2. zadatak – 13 poena, 3. zadatak – 12 poena

Vreme izrade: 60 minuta