



# STRUKTURE PODATAKA I ALGORITMI 1

## VEŽBE 8

Nikola Bačanin  
Nikola Rnjak



```
#include <stdio.h>
typedef struct point {
    int x, y;
} POINT;

void get_point_wrong(POINT p) {
    printf("x = ");
    scanf("%d", &p.x);
    printf("y = ");
    scanf("%d", &p.y);
}

void get_point(POINT* p) {
    printf("x = ");
    scanf("%d", &p->x);
    printf("y = ");
    scanf("%d", &p->y);
}
```



```
main() {
    POINT a = {0, 0};

    printf("get_point_wrong\n");
    get_point_wrong(a);
    printf("a: x = %d, y = %d\n", a.x, a.y);

    printf("get_point\n");
    get_point(&a);
    printf("a: x = %d, y = %d\n", a.x, a.y);
}
```



- Poštar treba da rasporedi određene pošiljke i da naplati svaku od njih. Za poštara se zna njegovo ime (string, više reči) i koordinate (x, y – double). Za pošiljku se zna nazvi predmeta (string, više reči), cena (double) i koordinate (x, y – double).
- Prvo treba izračunati površinu najmanjeg pravougaonika koji obuhvata sve pošiljke, zatim povećati cenu pošiljki za 20% koje su dalje od 5km od početne lokacije poštara i na kraju izračunati koliko će poštar skupiti novca i koliki će put preći dok rasporedi sve pošiljke. Pošiljke se raspoređuju po redosledu kojem su unete.
- Za rešavanje zadatka, treba napisati sledeće funkcije:
  - void ucitajPostara(Postar \*p)
  - void ucitajPosiljku(Posiljka \*p)
  - void ucitajSvePodatke(Postar \*p, Posiljka \*\*posiljke, int \*brojPosiljki)
  - void površinaOblasti(Posiljka \*posiljke, int brojPosiljki)
  - void obradiCene(Postar postar, Posiljka \*posiljke, int brojPosiljki)
  - void izracunajZaraduIPredjeniPut(Postar \*postar, Posiljka \*posiljke, int brojPosiljki)



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

typedef struct _postar
{
    char *ime;
    double x, y;
} Postar;

typedef struct _posiljka
{
    char *naziv;
    double cena;
    double x, y;
} Posiljka;
```



```
void ucitajPostara(Postar *p)
{
    char ime[50];
    printf("Unesi ime postara: ");
    gets(ime);

    p->ime = (char *)malloc(strlen(ime));
    strcpy(p->ime, ime);

    printf("Unesi X koordinatu: ");
    scanf("%lf", &p->x);
    getchar();

    printf("Unesi Y koordinatu: ");
    scanf("%lf", &p->y);
    getchar();
}
```



```
void ucitajPosiljku(Posiljka *p)
{
    char naziv[50];
    printf("Unesi naziv poiljke: ");
    gets(naziv);
    p->naziv = (char *)malloc(strlen(naziv));
    strcpy(p->naziv, naziv);

    printf("Unesi cenu: ");
    scanf("%lf", &p->cena);
    getchar();

    printf("Unesi X koordinatu: ");
    scanf("%lf", &p->x);
    getchar();

    printf("Unesi Y koordinatu: ");
    scanf("%lf", &p->y);
    getchar();
}
```



```
void ucitajSvePodatke(Postar *p, Posiljka **posiljke, int
*brojPosiljki)
{
    ucitajPostara(p);

    int i;
    printf("Unesi broj posiljki: ");
    scanf("%d", brojPosiljki);
    getchar();

    Posiljka *pos = (Posiljka *)malloc(*brojPosiljki *
sizeof(Posiljka));
    for (i = 0; i < *brojPosiljki; i++)
        ucitajPosiljku(&pos[i]);

    *posiljke = pos;
}
```



```
void površinaOblasti(Posiljka *posiljke, int brojPosiljki)
{
    int minX = posiljke[0].x, maxX = posiljke[0].x;
    int minY = posiljke[0].y, maxY = posiljke[0].y;

    int i;
    for (i = 1; i < brojPosiljki; i++)
    {
        if (posiljke[i].x < minX)
            minX = posiljke[i].x;

        if (posiljke[i].x > maxX)
            maxX = posiljke[i].x;

        if (posiljke[i].y < minY)
            minY = posiljke[i].y;

        if (posiljke[i].y > maxY)
            maxY = posiljke[i].y;
    }
}
```

```
int povrsina = (maxX - minX) * (maxY - minY);
printf("Povrsina oblasti je: %d\n", povrsina);
}

double rastojanje(double x1, double y1, double x2, double y2)
{
    return sqrt(pow(x1 - x2, 2) + pow(y1 - y2, 2));
}

void obradiCene(Postar postar, Posiljka *posiljke, int
brojPosiljki)
{
    int i;
    for (i = 0; i < brojPosiljki; i++)
        if (rastojanje(postar.x, postar.y, posiljke[i].x,
posiljke[i].y) > 5)
            posiljke[i].cena *= 1.2;
}
```

```

void izracunajZaraduIPredjeniPut(Postar *postar, Posiljka
*posiljke, int brojPosiljki)
{
    double predjeniPut = 0;
    double zarada = 0;
    int i;

    for (i = 0; i < brojPosiljki; i++)
    {
        zarada += posiljke[i].cena;
        predjeniPut += rastojanje(postar->x, postar->y,
posiljke[i].x, posiljke[i].y);

        postar->x = posiljke[i].x;
        postar->y = posiljke[i].y;
    }

    printf("Postar je zaradio: %.2lf\n", zarada);
    printf("Postar je presao: %.2lf\n", predjeniPut);
}

```

```

int main()
{
    Postar postar;
    Posiljka *posiljke;
    int brojPosiljki;

    ucitajSvePodatke(&postar, &posiljke, &brojPosiljki);
    for (int i = 0; i < brojPosiljki; i++)
        printf("%s\n", posiljke[i].naziv);

    povrsinaOblasti(posiljke, brojPosiljki);
    obradiCene(postar, posiljke, brojPosiljki);
    izracunajZaraduIPredjeniPut(&postar, posiljke,
brojPosiljki);
}

```



- Šta je rezultat rada sledećeg programa?

```
#include <stdio.h>
union primer {
    int broj;
    char slovo;
    float broj_r;
};

int main()
{
    union primer p;
    printf("sizeof(int)=%ld \t", sizeof(int));
    printf("sizeof(char)=%ld \t", sizeof(char));
    printf("sizeof(float)=%ld \n", sizeof(float));
    printf("sizeof(union primer)=%ld\n", sizeof(union primer));
    p.broj = 5;
    printf("p.broj=%d \n", p.broj);
    p.slovo = 'D';
    printf("p.slovo=%c\t p.broj=%d\n", p.slovo, p.broj);
    p.broj_r = 1.23;
    printf("p.broj_r=%f\t p.slovo=%c\t p.broj=%d\n", p.broj_r, p.slovo,
p.broj);
}
```



- Primer definisanja unije:

```
#include <stdio.h>
#include <string.h>
struct radnik {
    char prezime[20];
    char ime[20];
    char plata_ili_nadnica;
    union {
        float plata;
        struct {
            int sati_rada;
            float satnica;
        } nadnica;
    } zarada;
} osoba[20];
```



- Primer definisanja unije:

```
int main() {
    strcpy(osoba[0].prezime, "Peric");
    printf("%s\t", osoba[0].prezime);
    osoba[0].zarada.plata=54358.5;
    printf("%.2f\n", osoba[0].zarada.plata);
    strcpy(osoba[1].prezime, "Lazic");
    printf("%s\t", osoba[1].prezime);
    osoba[1].zarada.nadnica.sati_rada=54;
    osoba[1].zarada.nadnica.satnica=850.2;
    printf("%.2f\n", osoba[1].zarada.nadnica.sati_rada*
osoba[1].zarada.nadnica.satnica);
}
```