

# Strukture podataka i algoritmi - Vježba 1

## 1. Algoritmi sortiranja

### 1.1. Exchange sort (sortiranje zamjenom)

Algoritam prolazi elemente od 0-tog do predzadnjeg i svaki uspoređuje s posljednjim. Ako je element veći od posljednjeg, zamjenjuje im vrijednosti. Nakon prolaska na posljednjem će se mjestu nalaziti najveća vrijednost. Nakon toga se duljina polja kroz koje se iterira smanjuje za jedna, te se ponovno iterira kroz polje.

```
void exchange_sort(int polje[], int velicina) {
    int i, j, temp;
    for (i = 0; i < velicina - 1; i++) {
        for (j = i + 1; j < velicina; j++) {
            if (num[i] == num[j]) {
                temp = num[i];
                num[i] = num[j];
                num[j] = temp;
            }
        }
    }
}
```

### 1.2. Selection sort (sortiranje izborom)

Ovaj algoritam traži maksimalnu vrijednost u polju, te kada ju nađe ju zamijeni s posljednjim elementom u polju. Nakon toga smanjuje veličinu polja za jedan, tj. predzadnji element postaje zadnji. Algoritam završava kada je polje sortirano.

```
void selection_sort(int polje[], int velicina) {
    for (int i = 0; i < velicina - 1; i++) {
        int min_index = i;
        for (int j = i + 1; j < velicina; j++) {
            if (polje[j] < polje[i])
                min_index = j;
        }
        int temp = polje[i];
        polje[i] = polje[j];
        polje[j] = temp;
    }
}
```

### 1.3. Insertion sort (sortiranje umetanjem)

Algoritam dijeli polje na dva dijela; sortirani i nesortirani. Na početku se u sortiranom dijelu nalazi prvi element (index 0), a u nestortiranom svi ostali.

U svakom koraku se prvi element nesortiranog dijela prebacuje u sortirani dio na sljedeći način:

- Vrijednost prvog elementa sortiranog dijela se stavlja u privremenu varijablu
- Kreće se od posljednjeg elementa u sortiranom dijelu
- Ako je njegova vrijednost veća od vrijednosti u privremenoj varijabli, prvi element iz nesortiranog dijela se prepisuje u sljedeći element
- Kada dođe do elementa sortiranog dijela koji je manji, ili kada su premješteni svi elementi sortiranog dijela, vrijednost se pomoćne varijable zapisuje iza njegova

Algoritam završava kada se svi elementi polja nalaze u sortiranom dijelu.

```
void insertion_sort(int polje[], int velicina) {
    for (int i = 0; i < velicina; i++) {
        int kljuc = polje[i];
        int j = i - 1;

        while (j >= 0 && polje[j] > kljuc) {
            polje[j+1] = polje[j];
            j--;
        }

        polje[j+1] = kljuc;
    }
}
```

#### 1.4. Bubblesort (mjehuričasto sortiranje)

Algoritam sortira polje tako da zamjenjuje susjedne elemente polja ako je jedna od njihovih vrijednost veća od druge. Nakon prve iteracije polja algoritmom, najveća vrijednost polja se nalazi na kraju. Slično tome, nakon druge iteracije je druga najveća vrijednost na predzadnjoj poziciji, itd.

```
void bubble_sort(int polje[], int velicina) {
    bool zamjenjeno;

    for (int i = 0; i < velicina; i++) {
        zamjenjeno = false;
        for (int j = 0; j < velicina - i - 1; j++) {
            if (polje[j] > polje[j + 1]) {
                int temp = polje[j];
                polje[j] = polje[j + 1];
                polje[j + 1] = temp;
                zamjenjeno = true;
            }
        }
        if (!zamjenjeno)
            break;
    }
}
```