# STRUKTURE PODATAKA I ALGORITMI

ZBIRKA ZADATAKA

Ishod učenja 3

2024-2025.

## Zadatak 1

Napravite vektor koji sadrži sve brojeve iz raspona [1901, 2000], ali nasumično razbacane. Nakon toga, koristeći funkcije za ručno uređivanje i održavanje vektora kao hrpe, napravite sljedeće korake:

- Prikažite sve brojeve iz hrpe silaznim redoslijedom.
- Dodajte brojeve iz raspona [55, 99] u hrpu.
- Opet, prikažite sve brojeve iz hrpe silaznim redoslijedom.

*Moguće rješenje:*

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    vector<int> v;
    for (int i = 1901; i <= 2000; i++) {
        v.push_back(i);
    }

    random_shuffle(v.begin(), v.end());

    // Napravimo hrpu u O(n)
    make_heap(v.begin(), v.end());

    // Prikažemo sve brojeve iz hrpe silaznim redoslijedom u O(logn).
    while (!v.empty()) {
        cout << v[0] << " ";

        pop_heap(v.begin(), v.end());
        v.pop_back();
    }
    cout << endl << endl;

    // Dodamo brojeve iz raspona [55, 99] u hrpu u O(logn).
    for (int i = 55; i <= 99; i++) {
        v.push_back(i);
        push_heap(v.begin(), v.end());
    }

    // Prikažemo sve brojeve iz hrpe silaznim redoslijedom.
    while (!v.empty()) {
        cout << v[0] << " ";

        pop_heap(v.begin(), v.end());
        v.pop_back();
    }
    cout << endl << endl;
}
```

## Zadatak 2

Napišite program koji omogućuje obradu više objekata tipa Pismo prema redoslijedu prioriteta. Neka svaki objekt tipa Pismo sadrži ime primatelja, adresu primatelja i prioritet, gdje string "LOWEST" označava najniži prioritet, "MEDIUM" srednji prioritet, a "HIGHEST" najviši prioritet. Napravite pet objekata tipa Pismo i pomoću prioritetnog reda ispišite ih po prioritetu (od najvišeg do najnižeg).

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <queue>
using namespace std;

struct Pismo {
    string ime_primatelja;
    string adresa_primatelja;
    string prioritet;

    Pismo(string ime_primatelja, string adresa_primatelja, string prioritet) {
        this->ime_primatelja = ime_primatelja;
        this->adresa_primatelja = adresa_primatelja;
        this->prioritet = prioritet;
    }
};

struct MojKomparator {
    bool operator() (Pismo& o1, Pismo& o2) {
        if (o1.prioritet == "HIGHEST") {
            return false;
        }
        else if (o2.prioritet == "HIGHEST") {
            return true;
        }
        else if (o1.prioritet == "MEDIUM") {
            return false;
        }
        else if (o2.prioritet == "MEDIUM") {
            return true;
        }
        else if (o1.prioritet == "LOWEST") {
            return false;
        }
        else if (o2.prioritet == "LOWEST") {
            return true;
        }
        return false;
    }
};

int main() {
    priority_queue<Pismo, vector<Pismo>, MojKomparator> pq;
    pq.push({ "Janko Strizic", "Blatni put 19A", "LOWEST" });
    pq.push({ "Pero Petricevic", "Zupanijska cesta 7", "MEDIUM" });
    pq.push({ "Ranko Ramic", "Plava ulica 142", "MEDIUM" });
    pq.push({ "Albert Peric", "Ulica Mire Mirica 1", "HIGHEST" });
```

```
    pq.push({ "Branka Brankic", "Crveni vijenac 5", "LOWEST" });

    while (!pq.empty()) {
        cout << pq.top().ime_primatelja << " " << pq.top().adresa_primatelja << " ("
<< pq.top().prioritet << ")" << endl;
        pq.pop();
    }

}
```

## Zadatak 3

Napravite vektor koji sadrži sve brojeve iz raspona [1, 100], ali nasumično razbacane. Nakon toga, koristeći funkcije za ručno uređivanje i održavanje vektora kao hrpe, napravite sljedeće korake:

- Prikažite sve brojeve iz hrpe silaznim redoslijedom.
- Dodajte brojeve iz raspona [1, 10] u hrpu.
- Opet, prikažite sve brojeve iz hrpe silaznim redoslijedom.

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    vector<int> v;
    for (int i = 1; i <= 100; i++) {
        v.push_back(i);
    }

    random_shuffle(v.begin(), v.end());

    make_heap(v.begin(), v.end());

    while (!v.empty()) {
        cout << v[0] << " ";

        pop_heap(v.begin(), v.end());
        v.pop_back();
    }
    cout << endl << endl;

    for (int i = 1; i <= 10; i++) {
        v.push_back(i);
        push_heap(v.begin(), v.end());
    }

    while (!v.empty()) {
        cout << v[0] << " ";

        pop_heap(v.begin(), v.end());
        v.pop_back();
    }
    cout << endl << endl;
}
```

## Zadatak 4

Napišite program koji omogućuje obradu niza IpPacket objekata prema prioritetima. Neka svaki objekt tipa IpPacket sadrži izvornu IP adresu, odredišnu IP adresu i prioritet izražen kao broj od 1 do 300:

- Svi brojevi u rasponu [1, 100] označavaju najniži mogući prioritet.
- Svi brojevi u rasponu [101, 200] označavaju srednji prioritet.
- Svi brojevi u rasponu [201, 300] označavaju najviši mogući prioritet.

Nije bitno koji je točan broj, samo je bitna njegova pripadnost rasponu (npr. brojevi 1 i 100 označavaju isti, najniži prioritet). Napravite pet objekata tipa IpPacket i pomoću prioritetnog reda ih prikažite po prioritetu (od najvišeg do najnižeg).

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <queue>
using namespace std;

struct IpPacket {
    string izvorisna;
    string odredisna;
    int prioritet;
    int prioritet_klasa;

    IpPacket(string izvorisna, string odredisna, int prioritet) {
        this->izvorisna = izvorisna;
        this->odredisna = odredisna;
        this->prioritet = prioritet;

        if (prioritet <= 100) {
            prioritet_klasa = 1;
        }
        else if (prioritet <= 200) {
            prioritet_klasa = 2;
        }
        else {
            prioritet_klasa = 3;
        }
    }
};

struct MojKomparator {
    bool operator() (IpPacket& o1, IpPacket& o2) {
        if (o1.prioritet_klasa < o2.prioritet_klasa) {
            return true;
        }
        return false;
    }
};

int main() {
    priority_queue<IpPacket, vector<IpPacket>, MojKomparator> pq;
    pq.push({ "192.168.1.1", "192.168.1.199", 15 });
    pq.push({ "192.168.1.25", "192.168.1.13", 102 });
    pq.push({ "192.168.1.1", "192.168.1.8", 168 });
```

```
    pq.push({ "192.168.1.1", "192.168.1.9", 299 });
    pq.push({ "192.168.1.1", "192.168.1.10", 85 });

    while (!pq.empty()) {
        cout << pq.top().izvorisna << " " << pq.top().odredisna << " (" <<
pq.top().prioritet << ")" << endl;
        pq.pop();
    }

}
```

## Zadatak 5

Učitajte sva imena iz datoteke **female_names.txt** u vektor. Nakon toga, koristeći funkcije za ručno uređivanje i održavanje vektora kao hrpe, organizirajte vektor u hrpu i ispišite sva imena padajućim abecednim redoslijedom.

*Moguće rješenje:*

```
#include <iostream>
#include <string>
#include <vector>
#include <fstream>
#include <algorithm>
using namespace std;

int main() {
    vector<string> v;
    string ime;

    ifstream dat("female_names.txt");
    if (!dat) {
        cout << "Greska pri otvaranju datoteke" << endl;
        return 1;
    }

    while (getline(dat, ime)) {
        v.push_back(ime);
    }

    make_heap(v.begin(), v.end());

    while (!v.empty()) {
        cout << v[0] << endl;

        pop_heap(v.begin(), v.end());
        v.pop_back();
    }

    dat.close();
}
```

## Zadatak 6

Napišite program koji omogućuje obradu niza IpPacket objekata prema redoslijedu prioriteta. Neka svaki objekt tipa IpPacket sadrži izvornu IP adresu, odredišnu IP adresu i prioritet kao niz "MAT" (najniži prioritet), "KES" (srednji prioritet) ili "KOR" (najviši prioritet). Napravite pet objekata tipa IpPacket i pomoću prioritetnog reda prikažite ih po prioritetu (od najvišeg do najnižeg).

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <queue>
using namespace std;

struct IpPacket {
    string izvorisna;
    string odredisna;
    string prioritet;

    IpPacket(string izvorisna, string odredisna, string prioritet) {
        this->izvorisna = izvorisna;
        this->odredisna = odredisna;
        this->prioritet = prioritet;
    }
};

struct MojKomparator {
    bool operator() (IpPacket& o1, IpPacket& o2) {
        if (o1.prioritet == "KOR") {
            return false;
        }
        else if (o2.prioritet == "KOR") {
            return true;
        }
        else if (o1.prioritet == "KES") {
            return false;
        }
        else if (o2.prioritet == "KES") {
            return true;
        }
        return false;
    }
};

int main() {
    priority_queue<IpPacket, vector<IpPacket>, MojKomparator> pq;
    pq.push({ "192.168.1.1", "192.168.1.199", "MAT" });
    pq.push({ "192.168.1.25", "192.168.1.13", "KES" });
    pq.push({ "192.168.1.1", "192.168.1.8", "KES" });
    pq.push({ "192.168.1.1", "192.168.1.9", "KOR" });
    pq.push({ "192.168.1.1", "192.168.1.10", "MAT" });

    while (!pq.empty()) {
        cout << pq.top().izvorisna << " " << pq.top().odredisna << " (" <<
pq.top().prioritet << ")" << endl;
        pq.pop();
```

```
    }
}
```

## Zadatak 7

Napravite vektor i osigurajte da je organiziran kao hrpa koristeći funkcije za ručno uređivanje i održavanje vektora kao hrpe. Omogućite korisniku sljedeće funkcionalnosti:

- Unos tri nova broja u hrpu.
- Ispis pet brojeva s hrpe (ako nema pet, ispišite koliko ima).
- Završetak programa.

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    vector<int> v;
    int broj;
    char opcija;

    cout << "A = unos brojeva u hrpu" << endl;
    cout << "D = ispis brojeva s hrpe" << endl;
    cout << "X = kraj" << endl;

    do {
        cout << "> ";
        cin >> opcija;

        switch (opcija) {
        case 'A':
            for (int i = 0; i < 3; i++) {
                cout << "# ";
                cin >> broj;
                v.push_back(broj);
                push_heap(v.begin(), v.end());
            }
            break;
        case 'D':
            for (int i = 0; i < 5; i++) {
                if (!v.empty()) {
                    cout << v[0] << endl;
                    pop_heap(v.begin(), v.end());
                    v.pop_back();
                }
            }
            break;
        }
    } while (opcija != 'X');
}
```

## Zadatak 8

Napišite program koji omogućuje obradu niza Osoba objekata abecedno prema prezimenu, a zatim prema imenu. Neka svaki objekt tipa Osoba sadrži ime i prezime. Napravite pet objekata tipa Osoba i pomoću prioritetnog reda ih prikažite abecedno prema prezimenu, a zatim prema imenu.

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <queue>
using namespace std;

struct Osoba {
	string ime;
	string prezime;

	Osoba(string ime, string prezime) {
		this->ime = ime;
		this->prezime = prezime;
	}
};

struct MojKomparator {
	bool operator() (Osoba& o1, Osoba& o2) {
		if (o1.prezime < o2.prezime) {
			return false;
		}
		else if (o1.prezime > o2.prezime) {
			return true;
		}
		else {
			if (o1.ime < o2.ime) {
				return false;
			}
			else if (o1.ime > o2.ime) {
				return true;
			}
		}
	}
};

int main() {
	priority_queue<Osoba, vector<Osoba>, MojKomparator> pq;
	pq.push({ "Petra", "Pranic" });
	pq.push({ "Ana", "Pranic" });
	pq.push({ "Violeta", "Violetic" });
	pq.push({ "Zana", "Pranic" });
	pq.push({ "Ana", "Anic" });

	while (!pq.empty()) {
		cout << pq.top().prezime << ", " << pq.top().ime << endl;
		pq.pop();
	}
}
```

## Zadatak 9

Napravite vektor i osigurajte da je organiziran kao hrpa koristeći funkcije za ručno uređivanje i održavanje vektora kao hrpe. Omogućite korisniku sljedeće funkcionalnosti:

- Dodavanje 10 slučajnih brojeva [1, 100] u hrpu.
- Ispis svih brojeva s hrpe.
- Završetak programa.

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

int main() {
    srand(time(nullptr));

    vector<int> v;
    char opcija;

    cout << "A = unos brojeva u hrpu" << endl;
    cout << "D = ispis brojeva s hrpe" << endl;
    cout << "X = kraj" << endl;

    do {
        cout << "> ";
        cin >> opcija;

        switch (opcija) {
        case 'A':
            for (int i = 0; i < 10; i++) {
                v.push_back(1 + rand() % (100 - 1 + 1));
                push_heap(v.begin(), v.end());
            }
            break;
        case 'D':
            while (!v.empty()) {
                cout << v[0] << endl;
                pop_heap(v.begin(), v.end());
                v.pop_back();
            }
            break;
        }
    } while (opcija != 'X');
}
```

## Zadatak 10

Napišite program koji omogućuje obradu niza objekata tipa HotelskaSoba. Neka svaki objekt tipa HotelskaSoba sadrži broj kata i broj sobe. Napravite pet objekata tipa HotelskaSoba i pomoću prioritetnog reda ih prikažite tako da prvo prikažete sobe s najvišeg kata, a zatim prema nižim katovima. Sve sobe s istog kata treba prikazati rastuće prema broju sobe.

*Moguće rješenje:*

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <queue>
using namespace std;

struct HotelskaSoba {
	int kat;
	int broj_sobe;

	HotelskaSoba(int kat, int broj_sobe) {
		this->kat = kat;
		this->broj_sobe = broj_sobe;
	}
};

struct MojKomparator {
	bool operator() (HotelskaSoba& o1, HotelskaSoba& o2) {
		if (o1.kat < o2.kat) {
			return true;
		}
		else if (o1.kat > o2.kat) {
			return false;
		}
		else {
			if (o1.broj_sobe < o2.broj_sobe) {
				return false;
			}
			else {
				return true;
			}
		}
	}
};

int main() {
	priority_queue<HotelskaSoba, vector<HotelskaSoba>, MojKomparator> pq;
	pq.push({ 2, 4 });
	pq.push({ 2, 3 });
	pq.push({ 3, 15 });
	pq.push({ 3, 14 });
	pq.push({ 1, 1 });

	while (!pq.empty()) {
		cout << pq.top().kat << pq.top().broj_sobe << endl;
		pq.pop();
	}
}
```