**SVEUČILIŠTE U RIJECI  
FAKULTET INFORMATIKE I DIGITALNIH TEHNOLOGIJA**

**Preddiplomski studij informatike**

**Seminarski rad – Završni ispit iz kolegija  
Algoritmi i strukture podataka**

**na temu**

**Algoritam ruksaka bez svojstva dijeljenja**

Autori: Ahmičić Dino  
 Kovačev Antonio  
Mentori: izv. prof. dr. sc. Marija Brkić Bakarić

mag. educ. math. et inf. Dejan Ljubobratović

U Rijeci, Rujan 2023.

Sadržaj

[1. Uvod 2](#_Toc145484637)

[1.1. Odabir algoritma 2](#_Toc145484638)

[2. Ruksak bez svojstva dijeljenja 3](#_Toc145484639)

[3. Program 4](#_Toc145484640)

[3.1. Učitavnja podataka 4](#_Toc145484641)

[3.2. Izrada algoritma 7](#_Toc145484642)

[3.3. Glavni izbornik 9](#_Toc145484643)

[4. Ispis programa 10](#_Toc145484644)

# Uvod

Kao završni rad kolegija algoritmi i strukture podataka, morali smo napraviti C++ program koji će uzimati sve podatke unutar excel tablice, priloženu kao .csv datoteka, te njima manipulirati na neki zadani način.

Za početak, dobili smo šest skupova zadataka (.csv datoteka), od kojih ćemo jednu morati koristiti kao izvor naših podataka. Način na koji smo odabrali našu datoteku jest taj da smo uzeli četiri zadnja broja JMBAG-a prvog studenta poredanog uzlazno po abecedi. JMBAG studenta Dino Ahmičić glasi 0069084637. Broj 4637 morali smo podijeliti sa 113 (ukupan broj stupaca svih skupova podataka), te ostatak dijeljenja uvećati za 1. Time dobijemo broj 5. Kako smo dobili broj 5, kao naš izbor podataka koristiti ćemo prvi skup podataka pod imenom „[*Cincinnati Car Crash Data Since 2010*](https://www.kaggle.com/datasets/steverusso/cincinnati-car-crash-data)“

Spomenuti skup podataka sadrži 27 stupaca, koji reprezentiraju podatke o prometnim nesrećama koje su se dogodile u gradu Cincinnati-ju od 2010. godine. Neke od informacija koje možemo dobiti iz ovoga skupa prometnih nesreća jesu datumi adresa lokacije nesreće, geografska širina i dužina lokacije, susjedstvo u kojem se desila nesreća, dob osobe koja je sudjelovala u nesreći, itd.

Kao glavni stupac podataka kojim ćemo manipulirati daljnjom izradom ovoga rada jest dob osobe koja je sudjelovala u nesreći, iz razloga što je to peti stupac unutar naše .csv datoteke.

## Odabir algoritma

Algoritam koji ćemo koristiti za manipulaciju podataka odabiremo na sličan način kao i kod odabira skupa podataka. Zadnje četiri znamenke JMBAG-a drugog studenta tima (Antonio Kovačev) glase 6261. Taj broj podijelimo sa 7, ostatak dijeljenja uvećamo za 1. Kao rezultat dobili smo broj 4, i time i naš algoritam koji ćemo implementirati, ruksak bez svojstva dijeljenja.

# Ruksak bez svojstva dijeljenja

Ruksak bez svojstva dijeljenja ili 0/1 problem ruksaka klasični je optimizacijski problem u informatici i matematici. To je specifična varijacija problema s ruksakom gdje nam je dan skup predmeta, svaki s težinom i vrijednošću, te ruksak s maksimalnim kapacitetom težine. Cilj je odrediti kombinaciju predmeta koje treba staviti u ruksak tako da njihova ukupna težina ne premašuje kapacitet ruksaka, a ukupna vrijednost je maksimizirana.

Glavne karakteristike algoritma jesu:

* Stavke – imamo skup stavki, svaka sa težinom i pripadnom vrijednošću.
* Kapacitet ruksaka – imamo ruksak maksimalnog kapaciteta težine kojeg nesmijemo prijeći
* Binarna odluka – možemo ili uzeti stavku (staviti je u ruksak) ili ju ostaviti

Cilj je pronaći podskup stavki (binarni vektor) koji maksimizira ukupnu vrijednost, a istovremeno osigurava da zbroj težina odabranih stavki ne premaši kapacitet ruksaka.

Vremenska složenost rješenja dinamičkog programiranja za problem ruksaka 0/1 obično je O(N \* T), gdje je N broj stavki, a T maksimalni kapacitet ruksaka. Ovaj algoritam pronalazi optimalno rješenje uzimajući u obzir sve moguće kombinacije stavki i njihovih težina kako bi se maksimizirala ukupna vrijednost uz poštivanje ograničenja težine.

# Program

## Učitavnja podataka

Za početak, preuzeli smo datoteku koju ćemo koristiti kao naš izvor podataka sa kaggle.com web stranice, te smo ju nazvali „Data.csv“ i spremili u direktorij našega C++ projekta.

Unutar naše skripte izradili smo jednu strukturu unutar koje ćemo spremati sve podatke iz datoteke. Svaka stavka unutar naše strukture reprezentirati će po jedan stupac naše datoteke.

A screen shot of a computer code

Description automatically generated

Slika 1. Glavna struktura

Inicijalizirajmo sada vektor strukture u koji ćemo spremati naše podatke.



Slika 2. Vektor strukture

Našu datoteku ćemo kao varijablu nazvati „*file*“, te ju čitamo funkcijom „*ifstream*“, a svaku ćeliju tablice uzimamo funkcijom „*getline()*“. Također je bitno izostaviti prvi redak, jer je on zaglavlje tablice.

A screen shot of a computer code

Description automatically generated

Slika 3. Funkcije za čitanje datoteke

*While* petljom započinjemo čitanje, te koristimo „*istringstream*“ za procesiranje stringova unutar svake varijable. Deklariramo varijablu podatak koji reprezentira svaku instancu unutar svake ćelije, deklariramo inicijaliziramo instancu strukture, te varijablu index tipa integer.

A screen shot of a computer code

Description automatically generated

Slika 4. Početak čitanja datoteke

Hodanje kroz tablicu izvesti ćemo „*switch()*“ naredbom preko varijable „*index*“ koja će reprezentirati case-ove unutar switch-a. Svaki case pokazuje na posebni stupac tablice, te svakom interacijom kroz ćelije, spremaju se podaci u prethodno inicijaliziranu strukturu. Svakom iteracijom, varijabla index se povećava za 1, sve dok se ne dođe do kraja datoteke. Svakom iteracijom kroz redove, „*push\_back()*“ funkcijom se podaci spremaju unutar vektora. Nakon spremanja svih podataka, funkcijom „*file.close()*“ zatvaramo datoteku.

A screenshot of a computer program

Description automatically generated

Slika 5. Switch() funkcija

Kako smo prethodno dijeljenjem znamenki 6261 dobili broj 5, za manipulaciju podataka koristiti ćemo peti stupac naše tablice (dobi osoba u prometnim nesrećama). Radi jednostavnosti, uzeti ćemo po jednu vrijednost svake dobi, te ih poredati uzlazno. U našem slučaju, imati ćemo ukupno 24 različitih vrijednosti.

A screen shot of a computer code

Description automatically generated

Slika 6. Funkcija za ispis dobi sudionika nesreća



Slika 7. Izlaz funkcije

## Izrada algoritma

Radi jednostavnosti, pretvorili smo vektor vrijednosti svih dobi u skup. Isto tako, kako algoritam zahtjeva i težine svih vrijednosti, inicijalizirali smo još jedan skup u kojem smo unesli sve vrijednosti iz vektora vrijednosti, uvećanih za 1.

A white screen with blue text

Description automatically generated

Slika 8. Skupovi

Naša funkcija, tipa int, prima 3 parametra, skup vrijednosti, skup tezina stavki, te integer vrijednost kapaciteta ruksaka.  
Definiramo dva vektora vrijednosti i težine kojima ćemo manipulirati, te varijablu tipa integer n koja nam govori koliko je stavki unutar vektora.  
Također, inicijalizirati ćemo 2D vektor koji ima n+1 redaka i kapacitet+1 stupaca, inicijalno ispunjen nulama. 2D vektor ćemo koristiti sa spremanje rezultata potproblema algoritma.

A screen shot of a computer code

Description automatically generated

Slika 9. Početak algoritma

Sljedeće, if() naredbom provjeravamo da li je težina trenutne stavke jednaka ili manja od ukupnog kapaciteta ruksaka. Ako je, stavka je potencijalni kandidat za stavljanje u ruksak.

Računamo maksimalnu vrijednost koja se može dobiti pri trenutnom kapacitetu ruksaka *t* i trenutnoj stavki *i* uzimajući u obzir dvije mogućnosti:

* dp[i-1][t] 🡪 Maksimalna vrijednost dobivena bez uključivanja stavke
* dp[i - 1][t - tezina[i - 1]] + vrijednost[i – 1] 🡪 Maksimalna vrijednost dobivena uključivanjem stavke i

U slučaju kada trenutna stavka premašuje kapacitet naprtnjače, ona se odbacije, te se maksimalna vrijednost vraća sa prethodnom uključenom stavkom.

Te pri kraju algoritma, vraćamo maksimalnu dostignutu vrijednost.

A computer code with text

Description automatically generated

Slika 10. Kraj algoritma

## Glavni izbornik

Glavni izbornik koji će se korisniku prikazati smo izradili ovako:

A text on a white background

Description automatically generated

Slika 11. Glavni izbornik

Kada korisnik unese broj 1, prikazati će mu se sve vrijednosti dobi osoba stradalih u prometnim nesrećama.

Kada korisnik unese broj 2, prikazati će mu se sve vrijednosti težina koje smo prethodno odredili.

Kada korisnik unese broj 3, dobiva opciju unosa proizvoljnog kapaciteta ruksaka, s kojim će se izvoditi algoritam. Nakon unosa, algoritam odradi svoje, te se korisniku ispisuje maksimalna vrijednost moguća za problem ruksaka za zadane vrijednosti.

Izbornik smo implementirali funkcijom „*switch()“:*

A screenshot of a computer code

Description automatically generated

Slika 12. Switch() funkcija

# Ispis programa

A black screen with white text

Description automatically generated

Slika 13. Ispis pri pokretanju programa

A black screen with white text

Description automatically generated

Slika 14. Ispis prvog izbora

A black screen with white text

Description automatically generated

Slika 15. Ispis drugog izbora

A black screen with white text

Description automatically generated

Slika 16. Ispis trećeg izbora