# Kompresija u GZIP fajl formatu

Projekat podrazumeva kompresiju različitih fajlova u .gzip format i njihovo vraćanje u originalni format bez gubitaka podataka i kvaliteta.
Kompresija podrazumeva primenu DEFLATE algoritma koji se sastoji od
LZ77 i Huffman code algoritama primenjenih na fajl koji se kompresuje.
Nakon kompresije biće omogućeno vraćanje fajla u prethodni format primenom dekodovanja kompresovanog .gzip fajla.
Primena dekodovanja treba da vrati fajl u njegov originalni oblik.

Projekat će biti rađen u programskom jeziku Python, po potrebi uz pomoć nekih od njegovih biblioteka ( NumPy,... ).

Projekat izrađuje: Jelena Dokić

**Kompresija**:
Primenjuje se prvo LZ77 algoritam, a zatim se na njegove slogove primenjuje Huffmann code algoritam.

**LZ77**:
LZ77 kompresija pronalazi karaktere ili niz karaktera koji su se već pre nalazili u ulaznom fajlu i menja ih referencom na njihovu prvu pojavu. Referenca na prvu pojavu se sastoji od dva broja: daljine, koja govori koliko daleko nazad se taj niz karaktera pre toga pojavio i dužine, koja prikazuje koliko znakova iz prve pojave se nalazi u trenutnoj pojavi.

`Blah blah blah blah blah!` - ulazni string (uočavamo delove koji se ponavljaju)

`Blah blah b` - prvo ponavljanje dela `lah b`

`Blah b[D=5,L=5]` -kompresovana prva pojava

- D - 5 mesta u nazad
- L - 5 mesta od lokacije koju nalazimo pomoću D

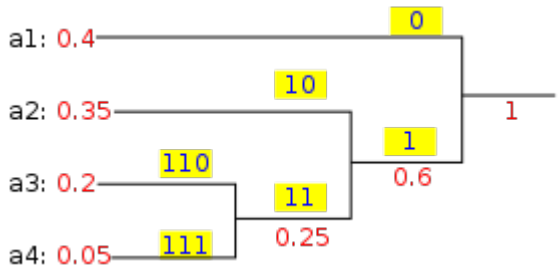
`Blah b[D=5, L=18]!` - kompresovan ceo ulazni string

Prethodne pojave koje se referenciraju se čuvaju u rečniku radi lakšeg indeksiranja:

`dictionary[...] = {index, character}`

- indeks predstavlja lokaciju u rečniku
- character je svaki karakter koji se pojavljuje u slogovima

**Huffmann code algoritam**:
Svakom slogu iz LZ77 rečnika se dodeljuje binarni kod. Kodovi su različite dužine,srazmerno učestalosti pojavljivanja slogova.




a1,a2,a3,a4 - slogovi iz LZ77 rečnika

Zbog najučestalijeg pojavljivanja a1 sloga,on dobija najkraci kod,dok se slogovi a3 i a4 pojavljuju redje pa zato dobijaju znatno duže kodove. Na osnovu ukupnog pojavljivanja i same dužine

pokazuje se da ovaj način kodiranja smanjuje dužinu zapisivanja celog ulaznog fajla, u odnosu na zapisivanje gde svaki slog ima kod jednake dužine:

jednaki kodovi: 4 * 2 = 8
Huffman code: 4 * ( 0.4 * 1 + 0.35 * 2 + ( 0.2 + 0.05 ) * 3 ) = 7.4

**Dekompresija**:
Zbog jedinstvene identifikacije slogova u oba algoritma, dekompresija se obavlja pronalazenjem slogova i vraćanjem u prvobitno stanje.

Oba algoritma će moći pojedinačno da se testiraju, ali u kombinaciji bi trebali da daju najefiaksnije rešenje. Test će se pozivati iz komandne linije, kako za kompresovanje, tako i za dekompresovanje.

Kompresija će biti isprobana na raznim vrstama formata fajlova i slika ( .jpg, .png, .odt, .pdf, ... ).

**Literatura**:

osnovna ideja o načinu funkcionisanja kompresije: https://www.youtube.com/watch?v=OtDxDvCpPL4

originalni izvorni kod algoritma .gzip kompresije: https://git.savannah.gnu.org/cgit/gzip.git/

Objašnjenje deflate algoritma:
http://www.zlib.net/feldspar.html,
http://www.infinitepartitions.com/art001.html

LZ77 algoritam:
https://en.wikipedia.org/wiki/LZ77_and_LZ78

Huffman coding algoritam:
https://en.wikipedia.org/wiki/Huffman_coding