# UVOD

Glavni namen in cilji diplomskega dela Kreiranje stisnjenih (zip) datotek je razumevanje delovanja t. i. stiskalnih algoritmov in kreiranje enostavnega programa za stiskanje datotek s pripadajočimi funkcionalnostmi, pregled zgodovine stiskalnih algoritmov in kako so ti nastali in možne dodelave oz. predelave stiskalnih algoritmov.

Metode izdelave naloge bodo najprej zgodovinski pregled in načini stiskanja datotek, programiranje enostavnega programa v programskem jeziku C# s pomočjo programa Visual Studio, z vsemi funkcionalnostmi ki so: stiskanje datotek, odpiranje stisnjene datoteke in ekstrahiranje (razširjanje) vsebine datoteke, enkripcija datotek s pomočjo gesla in kreiranje vizualnega uporabniškega vmesnika (GUI).

Izdelal bom tudi analizo in testiral hitrosti stiskanja datotek s pomočjo testnih orodij v okolju Visual Studio. Najbolj zanimive izseke iz kode bom tudi komentiral v nalogi, v priloge bom dodal celotno skripto programa s komentarji.

# TEORETIČNI DEL

V teoretičnem delu naloge, se posvečam zgodovini stiskalnih algoritmov, njihovi uporabi nekoč in izboljšavam danes. Navajam možnosti, načine uporabe in skozi skripte in komentarje tudi funkcionalnosti algoritmov.

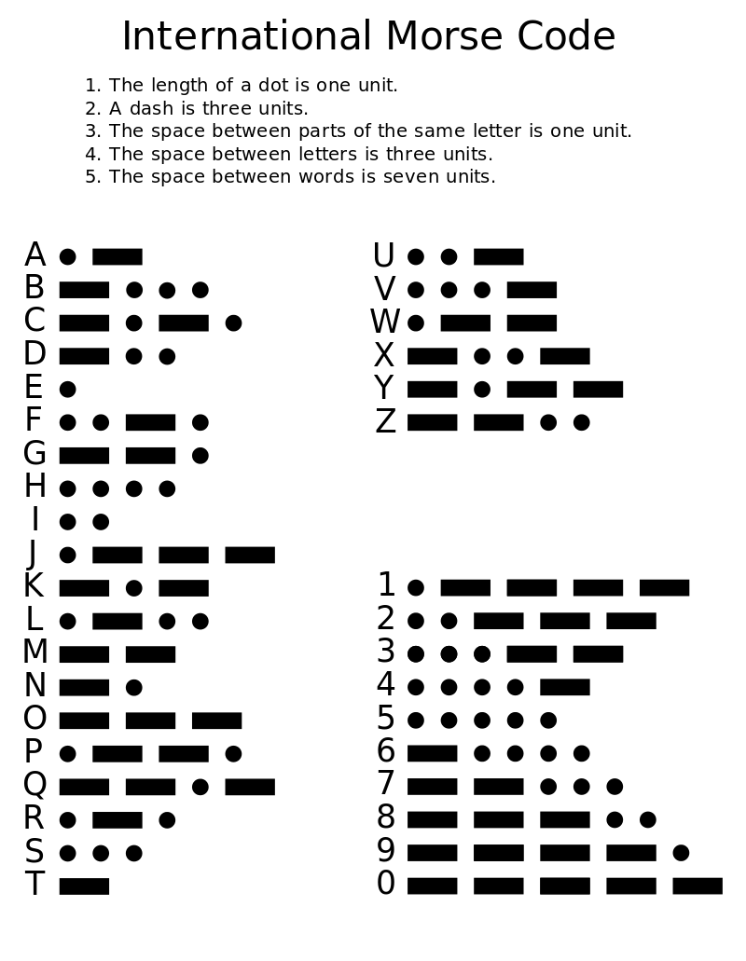
## ZGODOVINA

Kot prvi stiskalni algoritem se navaja že Morsejeva koda ali Morsejeva abeceda. Razvila sta jo izumitelj telegrafa Samuel Morse in Alfred Vail, leta 1835. Leta 1847 jo je Friedrich Clemens Gerke izpopolnil v obliko, kot jo poznamo danes.

Internacionalna Morsejeva abeceda je sestavljena iz šestindvajsetih angleških črk od A do Z, nekaj ne-angleških črk, postopkovnih znakov (ang. prosigns) in arabskih številk. Vsebuje samo velike črke in uporablja kombinacije črt in pik. Črte v znakih so trikrat daljše od pik, med vsako piko oz. črto je presledek v dolžini pike, med vsako črko v besedi je presledek v dolžini treh pik in razmak med besedami je v dolžini sedem pik.

Morsejeva abeceda je bila aktivno v uporabi v mornarici in oboroženih silah do leta 1999. Še vedno jo uporabljajo taborniki in skavti kot način enostavne komunikacije.

Slika 1: Internacionalna Morsejeva abeceda



Vir: <https://en.wikipedia.org/wiki/Morse_code#/media/File:International_Morse_Code.svg>

Velik preobrat se je začel dogajati med leti 1940 in 1948 ko je Claude E. Shannon s svojo disertacijo »A Mathematical Theory of Communication« objavljeno v Bell System Technical Journal julija in oktobra 1948, postavil danes znano Teorijo informacij. Iz teorije informacij, danes poznamo statistično sklepanje, kriptografijo, komunikacijske mreže, termodinamiko, kvantno računanje itd.

Pozneje leta 1949 sta Shannon v Bell Labs in R. M. Fano na MIT-ju skoraj istočasno razvila Shannon-Fano metodo za kodiranje simbolov, ki je bila odvisna od vedenja verjetnosti pojavitve posameznega znaka v nekem sporočilu. Po podani verjetnosti, je bila skonstruirana tabela znakov, ki je imela podane značilnosti:

* Različni znaki imajo različno število bitov.
* Znaki simbolov majhne verjetnosti prikaza imajo večje število bitov in znaki simbolov večje verjetnosti prikaza imajo manjše število bitov.
* Čeprav so simboli različnih dolžin bitov, so lahko edinstveno dekodirani.

Dve leti pozneje, leta 1951 je David Huffman obiskoval študij Informacijske teorije na MIT-ju in obiskoval predavanja Roberta Fana. Fano je za opravljanje predmeta, dal skupini študentov možnost pisanja izpita ali izdelave seminarja. Huffman se je odločil za izdelavo seminarja na temo iskanja najučinkovitejše metode binarnega kodiranja. Po nekajmesečnem raziskovanju in brez rešitev, je bil Huffman pripravljen zaključiti z delom in se pripraviti na končni izpit. Ravno takrat je doživel razodetje in ugotovil zelo podobno a učinkovitejšo tehniko Shannon-Fano metode. Glavna razlika je v branju drevesa. Pri Shannon-Fano se drevo bere od zgoraj dol, pri Huffmanovem kodiranju pa od spodaj gor. Huffmanov algoritem je bila prva implementacija algoritma, ki je vsebovala minimalno redundanco (izgubo) podatkov. Hitro je postalo jasno, da ima Huffmanov algoritem najboljše možne rezultate stiskanja.

Prve izvedbe Shannon-Fano in Huffman kodiranja so bile izvedene z uporabo strojne in »hardcoded«[[1]](#footnote-1) kode. Šele v 1970-ih letih, ko se je začel razvijati in uporabljati internet in spletna shramba programske opreme, so bile Huffmanove kode dinamično ustvarjene na podlagi vhodnih podatkov. Kasneje leta 1977 sta Abraham Lempel in Jacob Ziv razvila prelomni algoritem LZ77 imenovan tudi LZ1. To je bil prvi algoritem, ki je za kodiranje uporabil slovar, namesto drevesa. Natančneje, LZ77 je uporabljal dinamični slovar, s pomočjo protokola »Sliding window protocol«[[2]](#footnote-2), ki se v nekaterih implementacijah uporablja še danes. Več o tem v naslednjem poglavju.

Naslednje leto, leta 1978 sta Lempel in Ziv izboljšala svoj algoritem in objavila algoritem LZ78, imenovan tudi LZ2, ki pa je za razliko od LZ77 najprej razčlenil podatke in generiral statični slovar, namesto dinamičnega.

Oba algoritma LZ77 in LZ78 sta hitro pridobila na popularnosti, kar je vodilo do mnogih variant (glej sliko). Večina teh algoritmov je zamrla, le peščica se jih je ohranila do danes (LZMA, DEFLATE in LZX).

Leta 1984 je Terry Welch nadgradil LZ78 algoritem in objavil algoritem LZW, ki je izboljšal izvajanje algoritma LZ78 s tem, da je pohitril procese izvajanja na strojnem nivoju. Ta algoritem se je uporabljal in se še uporablja pri stiskanju slik v format GIF.

Večina standardnih algoritmov uporabljenih danes, temelji na algoritmu LZ77, ne zaradi tehnične superiornosti ampak zaradi tega, ker so algoritmi postali patentno obremenjeni, ko je korporacija Sperry leta 1984 patentirala LZW algoritem in začela tožiti prodajalce programske opreme, skrbnike strežnikov in celo končne uporabnike za uporabo GIF formata slik brez licence. Takrat je program za stiskanje v UNIX sistemih, ki je uporabljal modifikacijo LZW algoritma imenovanega LZC, bil kmalu ukinjen zaradi kršenja pravic patenta. Tudi drugi razvijalci na sistemu UNIX so začeli odstopati od uporabe LZW algoritma zaradi pojavitve odprtokodnih algoritmov.

Korporacije in druge večje organizacije, so uporabljale stiskanje podatkov od objave Lempel-Ziv algoritmov naprej, saj so imele vedno večje potrebe po shranjevanju podatkov. Ampak stiskanje podatkov ni dobilo večje prepoznavnosti in možnosti uporabe do pojava interneta v poznih 1980-ih, ko se je potreba po stiskanju podatkov eksponentno povečala. Pasovna širina pri internetu je bila omejena in draga, zato je stiskanje podatkov pomagalo ublažiti ozka grla pri prenosu podatkov. Stiskanje podatkov je postalo še kako zaželjeno, ko je internet dosegel vsakdanje uporabnike, ki so pošiljali, delili datoteke, slike ipd. Za izpolnitev teh potreb je bilo razvitih več novih arhivskih formatov kot so ZIP, GIF in PNG.

Thom Henderson je prvi komercialno uspešen arhivski format imenovan ARC, izdal leta 1986 prek svojega podjetja System Enhancement Associates (SEA). ARC je bil eden prvih programov in formatov, ki je lahko združeval in stiskal več datotek hkrati in bil odprtokoden. ARC je bil hibrid Huffmanovega in LZW algoritma.

Pozneje je Phil Katz s podjetjem PKWARE Inc. opazil priljubljenost ARC-a in prišel na idejo, da bo algoritem izboljšal s pisanjem rutin v strojnem jeziku. Leta 1987 je izdal svoj program PKARC kot shareware. SEA je PKWARE tožila zaradi kršenja blagovne znamke in avtorskih pravic. Sodišče je za primerjavo obeh programov imenovalo neodvisnega strokovnjaka za programsko opremo, Johna Navasa, ki je ugotovil, da je PKARC izpeljan del ARC-a, pri katerem so komentarji v obeh programih pogosto enaki, vključno s pravopisnimi napakami. Tožnik in tožene stranke so 2. avgusta 1988 napovedali poravnavo tožbe, ki je vključevala Zaupno pogodbo o navzkrižni licenci, po kateri je SEA licencirala PKWARE za vse programe, združljive z ARC, ki jih je PKWARE objavil v obdobju, ki se začne s prvo izdajo PKARC konec leta 1985 do 31. julija 1988 v zameno za nerazkrito plačilo. PKWARE je v sporazumu plačal SEA za pridobitev licence, ki je dovoljevala distribucijo programov, združljivih s PKWARE, do 31. januarja 1989, nato pa PKWARE ne bo licenciral, objavljal ali distribuiral programov ali pripomočkov, ki so združljivi z ARC. V zameno je PKWARE licenciral SEA za uporabo svoje izvorne kode za programe, združljive s PKWARE ARC. Podjetje PKWARE se je tudi strinjalo, da bo prenehalo uporabljati SEA zaščitni znak "ARC" in da bo spremenilo imena ali znamke, ki se uporabljajo v programih PKWARE, na nejasne oznake. Preostale podrobnosti sporazuma so bile zapečatene. Toženci pri doseganju poravnave niso priznali nobene krivde. S sklepom sodišča je bilo razvidno, da je bilo naloženo plačilo odškodnine tožniku zaradi dejanj toženca, ki kršijo tožnikove avtorske pravice, blagovno znamko in dejanja nepoštene trgovinske prakse in nepoštene konkurence.

Leta 1989 je Phil Katz zaradi tožbe predelal ARC algoritem in izdelal danes najbolj uporabljeni format ZIP. Format je na začetku uporabljal LZW algoritem ampak zaradi uporabe patenta, je lahko uKatz pozneje z izdajo programa PKZIP 2.0 LZW zamenjal za DEFLATE. Ta različica arhivske datoteke je v isti obliki prisotna še danes.

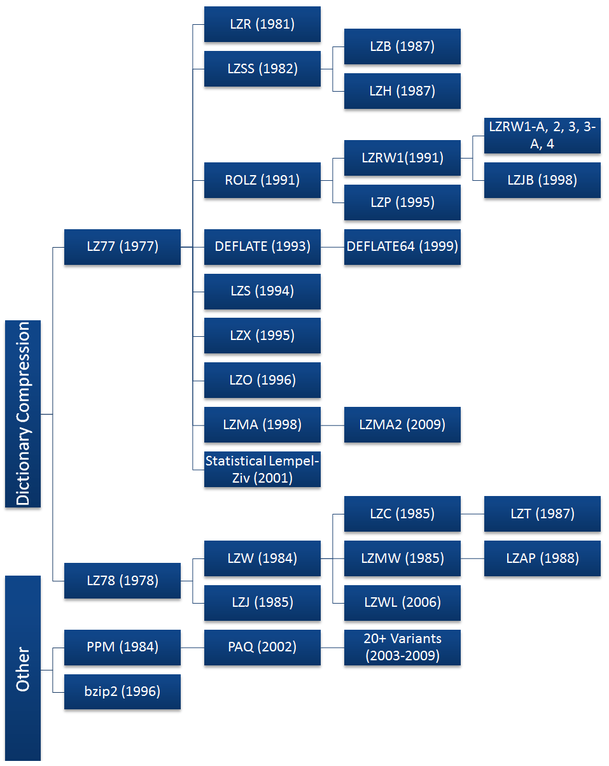
Format ZIP in drugi formati bazirani na DEFLATE algoritmu so bili na najboljši do sredine 1990-ih, ko so se začeli pojavljati novi in izboljšani formati. Leta 1993 je Eugene Roschal izdal program WinRAR, ki uporablja format RAR. Zadnja verzija RAR formata uporablja kombinacijo LZSS in PPM algoritma, za katera ni znano, kako sta se razvila. RAR je postal standardni format za pošiljanje datotek prek spleta, predvsem za distribucijo piratske programske opreme.

Odprtokodna implementacija algoritma Burrows-Wheeler transformacije imenovana BZIP2 predstavljena leta 1996 je hitro dobila na popularnosti na UNIX platformi in izpodbila DEFLATE baziran GZIP format. Še en odprtokodni program za stiskanje podatkov, ki se je pojavil leta 1999 je bil 7-Zip ali format 7z. 7-Zip bi lahko bil prvi format, ki je izzval prevlado ZIP in RAR formatov, zaradi na splošno visokega kompresijskega razmerja ter modularnosti. Ta oblika ni omejena na uporabo enega algoritma ampak lahko izbira med BZIP2, LZMA in PPMd algoritmi.

Zadnja najnovejša in najbolj sposobna različica stiskalnega algoritma je leta 2002 izdan algoritem imenovan PAQ, ki za delovanje uporablja statistične modele kontekstnega mešanja. Izdal ga je Matt Mahoney.

Prihodnost stiskalnih algoritmov je mogoče dokaj zanesljivo napovedati na podlagi trenutnih trendov, z uporabo PAQ algoritma in njegovih prihodnjih različic, ki dosega najvišja razmerja stiskanja. Algoritem Prediction by Partial Matching PPM oz. PPMd iz katerega izhaja PAQ, tudi lahko v prihodnosti vidi veliko izboljšav. Trenutno se je algoritem verige Lempel-Ziv-Markov oz. LZMA izkazal kot odličen kompromis med hitrostjo in visokim razmerjem stiskanja in je implementiran v večino datotek.

Slika 2: Diagram razvoja stiskalnih algoritmov

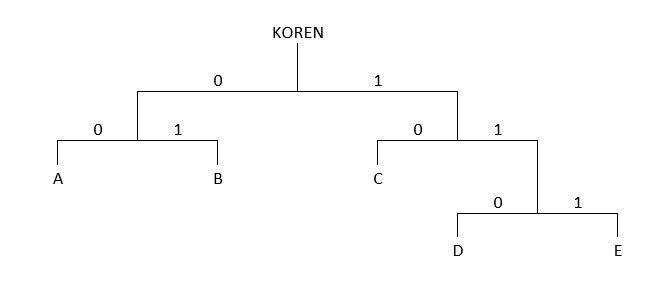


Vir: <https://ethw.org/History_of_Lossless_Data_Compression_Algorithms>, dostopno 1. 8. 2019

### Dizajn algoritmov in primeri kodiranja

Slika prikazuje enostavno Shannon-Fano drevo oblikovano za kodiranje ali dekodiranje enostavne simbolne abecede iz petih črk.

Slika 3: Enostavno Shannon-Fano drevo



Vir: Nelson, Mark, Gailly, Jean-loup (1996). The Data Compression Book, 2nd edition, M&T Books, str. 28. (Vir je preveden za potrebe naloge)

Tabela 1: Tabela kod abecede razbranih iz Shannon-Fano drevesa

|  |  |
| --- | --- |
| **Simbol** | **Koda** |
| A | 00 |
| B | 01 |
| C | 10 |
| D | 110 |
| E | 111 |

Vir: Nelson, Mark, Gailly, Jean-loup (1996). The Data Compression Book, 2nd edition, M&T Books, str. 28. (Vir je preveden za potrebe naloge)

### Metode kompresije

Kakšne metode kompresije smo poznali na začetku in kakšne metode poznamo danes in ali se je kaj spremenilo v zadnjih letih.

## ENKRIPCIJA

Pri stiskanju datotek je možna tudi enkripcija datotek, da zagotovimo njihovo varnost. Tukaj se bom posvetil različnim metodam in oblikam enkripcije in kako so te metode učinkovite pri varnosti datotek.

## IMPLEMENTACIJA

Tu bom na kratko opredelil orodja, ki obstajajo danes in kako so ta orodja implementirana v programe in aplikacije, ki jih uporabljamo.

# PRAKTIČNI DEL

Praktični del bo razdeljen na opis problema, ki ga imamo in to je programiranje enostavne aplikacije za stiskanje datotek, nato se bomo posvetili možni rešitvi in analizi/testiranju hitrosti programa.

## OPIS PROBLEMA

Ustvariti želimo enostavno konzolno aplikacijo z grafičnim uporabniškim vmesnikom za ustvarjanje ZIP in drugih stisnjenih datotek v C# okolju.

### Analiza trenutnih rešitev

Tukaj bom izdelal pregled vseh orodij in algoritmov, ki jih poznamo.

### Končne določbe

Naloga je določitev algoritmov in rešitev, ki jih bom uporabil.

## REŠITEV PROBLEMA

Pri rešitvi problema se bom najprej posvetil vizualni in programski zasnovi programa, kako bo program deloval.

### Programska zasnova

Pri programska zasnovi bom navedel strukturo programa in kako bo program deloval.

### Vizualna zasnova

Opis in skica zaslonskih mask programa in s katerim delom strukture bo posamezna zaslonska maska povezana.

### Stiskanje datotek

Rešitev stiskanja datotek in glavne značlnosti v programski kodi s komentarji.

### Stiskanje datotek z razdeljevanjem na več manjših datotek

Rešitev stiskanja datotek na več manjših in glavne značlnosti v programski kodi s komentarji.

### Ekstrahiranje (razširjanje) datotek

Rešitev ekstrahiranja (razširjanja) datotek in glavne značilnosti v programski kodi s komentarji.

### Odpiranje stisnjenih datotek za pregled vsebine

Rešitev odpiranja datotek in glavne značilnosti v programski kodi s komentarji.

### Enkripcija in dekripcija

Metoda in rešitev kriptiranja datotek

## UGOTOVITVE

Opis glavnih ugotovitev pri rešitvi problema in možne dodelave.

### Analiza in meritve

Analiza in meritve hitrosti izvajanja programa in primerjava z komercialnimi programi uporabljenimi vsakodnevno.

# SKLEPI

Ugotovitve, cilji in opredelitev razumevanja podanega problema.

# VIRI in LITERATURA

Podani viri.

# PRILOGE

Priložene bodo skripte celotnega programa.

1. “Hardcode” kodiranje programov je postopek programiranja, ki vključuje celotno kodo in potrebne podatke znotraj programa in ne iz zunanjih virov. (<https://en.wikipedia.org/wiki/Hard_coding>, dostopno 5. 8. 2019) [↑](#footnote-ref-1)
2. “Sliding window protocol” je značilnost paketnih protokolov za prenos podatkov, ki se uporabljajo kadar je potrebna zanesljiva oddaja paketov po vrstnem redu. Največ se uporablja pri internetnih povezavah v ISO/OSI podatkovnem modelu in pri TCP/IP povezavi. (<https://en.wikipedia.org/wiki/Sliding_window_protocol>, dostopno 5. 8. 2019) [↑](#footnote-ref-2)