Projekti iz predmeta Računarska elektronika

Aleksandra Lekić, Aleksandra Brkić

8. april 2017

Za svaki od zadataka potrebno je dostaviti:

- **Folder sa kodom programa koji zadovoljava traženu funkcionalnost. Ukoliko program generiše izlazne fajlove, potrebno je i njih priložiti.**
- **Izveštaj u kome se opisuje funkcija zadatka i kako je sam program realizovan. Na kraju izveštaja potrebno je dodati ceo kod projekta.**

Izveštaj i programe sačuvati u folderu sa imenom `REx`, gde je `x` broj projekta. Folder u vidu arhive poslati do 04.06.2017. mejlom sa *subject*-om `REx_projekat` na mejl adrese:

`lekic.aleksandra@etf.rs`
`aleksandra_brkic@live.com`

Odrbrana projekata će se organizovati po rasporedu koji će biti naknadno objavljen 14.06.2017.

Projekti označeni $*$ se mogu predati prvi dan ispitnog roka: junskog, julskog, septembarskog ili oktobarskog. Njihova odbrana će biti organizovana u danima pomenutog ispitnog roka po dogovoru.

Obrada slike

PGMA format slike

PGMA je *grayscale* ASCII format slike čiji je zapis:

```
P2
# pepper.ascii.pgm created by PGMA_IO::PGMA_WRITE.
256  256
255
26  47  49  43  44  44  46  41  43  42  43  43
42  42  41  39  44  39  42  45  46  42  39  39
42  42  41  39  44  39  42  45  46  42  39  39
36  39  37  37  35  41  37  35  33  34  33  37
...
```

Prvi red čini „magični broj" koji je u slučaju PGMA formata uvek P2. Zatim sledi karakter za novi red `0x0A` i red koji počinje sa `#`. Ovaj red predstavlja komentar. U sledećem redu je data širina slike odvojena sa dva razmaka i visina slike. Četvrti red predstavlja maksimalnu vrednost piksela slike, što je u slučaju 8-bitne slike 255. Od petog reda nadalje su date decimalne vrednosti piksela međusobno odvojene razmakom ili sa razmakom i znakom za novi red.

PGM slika se može prikazati korišćenjem programa *IrfanView* koji se može naći na linku: `http://www.irfanview.com/`. Detaljnije uputstvo je dato u datoteci `Obrada_slike` pod nazivom `pgm_files.pdf` u okviru fajlova za izradu projekta. Dato je i nekoliko slika PGMA formata koje se mogu koristiti za testiranje projekata.

Primer slike.

Na sajtu `http://people.sc.fsu.edu/~jburkardt/data/pgma/pgma.html` se može naći još primera slika, pri čemu ih je za upotrebu u programu potrebno dodatno umanjiti korišćenjem programa *IrfanView*.

Prikaz i promenu dimenzija slike moguće je uraditi i korišćenjem programa MATLAB, korišćenjem sledećih naredbi:

- Učitavanje i prikaz slike:

  ```
  pmga_img = imread('pepper.pgm');
  figure; imshow(pgma_img);
  ```

- Promena dimenzija slike:

  ```
  pgma_res = imresize(pgma_img,[a b]);
  ```

  Više informacija o načinu definisanja konstanti a i b može se dobiti pozivom komande `help imresize`.

- Čuvanje slike u PGMA formatu:

```
imwrite(pgma_res,'pepper_res.pgm','Encoding','ASCII');
```

  Jako je bitno specificirati encoding type prilikom čuvanja slike, jer ukoliko se on ne navede kao rezultat dobiće se binarni pgm fajl, odnosno PGMB format. Takođe, MATLAB generiše PGMA fajl tako što u jednom redu stoje, odvojeni razmacima, „magični broj", dimenzije slike i maksimalni intenzitet u slici, pa je radi konzistentnosti sa prethodnim opisom fajla potrebno korišćenjem nekog text editora, modifikovati fajl tako da ima formu shodno definisanim pravilima.

U narednih šest projekta potrebno je napraviti program koji po pokretanju pita za ime slike koju je potrebno obraditi. Nakon izvršene obrade slike, pita se za naziv izlazne slike i upisuju se obrađeni podaci.

Projekat 1 - skaliranje slike primenom bilinearne transformacije

Napisati program za skaliranje slike primenom bilinearne transformacije. Faktor s, sa kojim se vrši skaliranje, unosi se sa standardnog ulaza i uvek je veći ili jednak 1. Pored toga, sa standardnog ulaza na određeni način definiše se da li se vrši povećanje ili decimacija slike s puta.

.....

Povećanje dimenzija slike primenom bilinearne transformacije, vrši se na sledeći način:

Neka je $g(x', y')$ intenzitet piksela u skaliranoj slici na poziciji (x', y'). Koordinate u ulaznoj slici koje odgovaraju ovom pikselu izlazne slike su:

$$x = \frac{x'}{s} \text{ i } y = \frac{y'}{s}$$

Kako x i y ne moraju biti celi brojevi, intezitet piksela $(g(x', y'))$ dobija se posmatranjem intenziteta 4 najbliža piksela u originalnoj slici: $f(x_0, y_0)$, $f(x_0, y_0 + 1)$, $f(x_0 + 1, y_0)$ i $f(x_0 + 1, y_0 + 1)$, pri čemu je:

$$x_0 = \left\lfloor \frac{x'}{s} \right\rfloor \text{ i } x_r = x' - x_0\, s$$

$$y_0 = \left\lfloor \frac{y'}{s} \right\rfloor \text{ i } y_r = y' - y_0\, s$$

Sada se intenzitet piksela u izlaznoj slici računa upotrebom sledećih relacija:

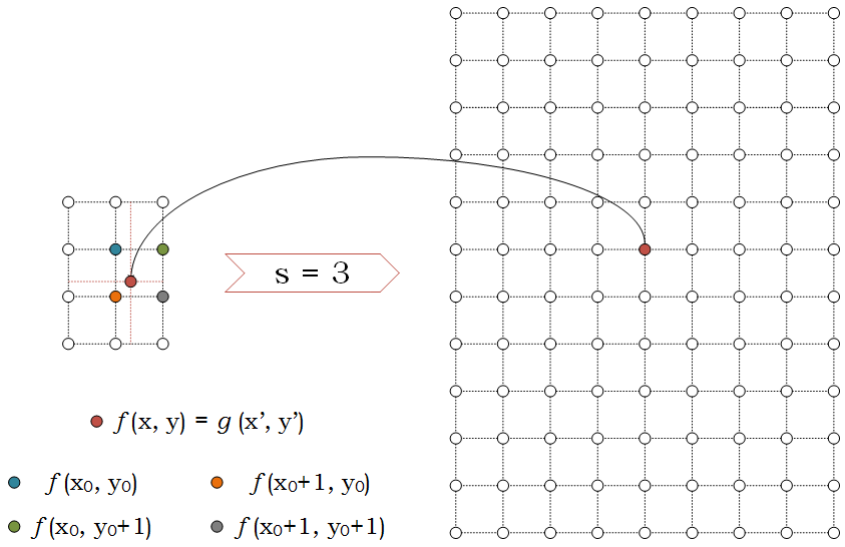
$$f(x_0, y) = \frac{f(x_0, y_0)\,(s - y_r) + f(x_0, y_0 + 1)\, y_r}{s}$$

$$f(x_0 + 1, y) = \frac{f(x_0 + 1, y_0)\,(s - y_r) + f(x_0 + 1, y_0 + 1)\, y_r}{s}$$

pa je:

$$g(x', y') = f(x, y) = \frac{f(x_0, y)\,(s - x_r) + f(x_0 + 1, y)\, x_r}{s}$$

Primer: Na slici 1 prikazan je jedan primer povećanja dimenzija slike u kome je faktor skaliranja $s = 3$. Označeni piksel u izlaznoj slici ima koordinate $(x', y') = (5, 4)$, pa su ekvivalentne koordinate ovog piksela u ulaznoj slici $(x, y) = (1.67, 1.33)$, odnosno $(x_0, y_0) = (1, 1)$ i $(x_r, y_r) = (2, 1)$. Neka je:



Slika 1: Primer skaliranja slike sa faktorom s=3

$$f(x_0, y_0) = 128,$$

$$f(x_0, y_0 + 1) = 150,$$

$$f(x_0 + 1, y_0) = 132 \text{ i}$$

$$f(x_0 + 1, y_0 + 1) = 145$$

Odavde je:

$$f(x_0, y) = \frac{128 \cdot (3-1) + 150 \cdot 1}{3} = \lfloor 135.33 \rfloor = 135,$$

$$f(x_0 + 1, y) = \frac{132 \cdot (3-1) + 145 \cdot 1}{3} = \lfloor 136.33 \rfloor = 136,$$

iz čega sledi da je:

$$g(x', y') = f(x, y) = \frac{135 \cdot (3-2) + 136 \cdot 2}{3} = \lfloor 135.667 \rfloor = 135$$

U slučaju decimacije slike dimenzije slike se određuju kao: $\left\lfloor \frac{N}{s} \right\rfloor$ i $\left\lfloor \frac{M}{s} \right\rfloor$ gde su M i N dimenzizije slike, a intenziteti piksela u izlaznoj slici određuju se prema formuli:

$$g(x', y') = f(sx', sy')$$

. . . . . .

Projekat 2 - skaliranje slike primenom metoda najbližeg suseda

Napisati program za skaliranje slike primenom metode najbližeg suseda. Faktor s, sa kojim se vrši skaliranje, unosi se sa standardnog ulaza i uvek je veći ili jednak 1. Pored toga, sa standardnog ulaza na određeni način definiše se da li se vrši povećanje ili decimacija slike s puta.

. . . . . .

Povećanje dimenzija slike primenom metoda najbližeg suseda, vrši se na sledeći način:

Neka je $g(x', y')$ intenzitet piksela u skaliranoj slici na poziciji (x', y'). Koordinate u ulaznoj slici koje odgovaraju ovom pikselu izlazne slike su:

$$x = \frac{x'}{s} \text{ i } y = \frac{y'}{s}$$

Kako x i y ne moraju biti celi brojevi, intezitet piksela ($g(x', y')$) dobija se posmatranjem 4 najbliža piksela u originalnoj slici: $f(x_0, y_0)$, $f(x_0, y_0 + 1)$, $f(x_0 + 1, y_0)$ i $f(x_0 + 1, y_0 + 1)$, pri čemu je:

$$x_0 = \left\lfloor \frac{x'}{s} \right\rfloor \text{ i } x_r = x' - x_0\, s$$

$$y_0 = \left\lfloor \frac{y'}{s} \right\rfloor \text{ i } y_r = y' - y_0\, s$$

Sada se intenzitet piksela u izlaznoj slici računa tako što se:

- ukoliko je $x_r < s - x_r$, uzima piksel u ulaznoj slici čija je x koordinata x_0, dok se u suprotnom uzima piksel čija je x koordinata $x_0 + 1$
- ukoliko je $y_r < s - y_r$, uzima piksel u ulaznoj slici čija je y koordinata y_0, dok se u suprotnom uzima piksel čija je y koordinata $y_0 + 1$

Primer: Na slici 1 prikazan je jedan primer povećanja dimenzija slike u kome je faktor skaliranja $s = 3$. Označeni piksel u izlaznoj slici ima koordinate $(x', y') = (5, 4)$, pa su ekvivalentne koordinate ovog piksela u ulaznoj slici $(x, y) = (1.67, 1.33)$, odnosno $(x_0, y_0) = (1, 1)$ i $(x_r, y_r) = (2, 1)$. Neka je:

$f(x_0, y_0) = 128$,

$f(x_0, y_0 + 1) = 150$,

$f(x_0 + 1, y_0) = 132$ i

$f(x_0 + 1, y_0 + 1) = 145$

Kako je $x_r > s - x_r$ i $y_r < s - y_r$, sledi da je:

$$g(x', y') = f(x, y) = f(x_0 + 1, y_0) = 132$$

U slučaju decimacije slike dimenzije slike se određuju kao: $\left\lfloor \frac{N}{s} \right\rfloor$ i $\left\lfloor \frac{M}{s} \right\rfloor$ gde su M i N dimenzije slike, a intenziteti piksela u izlaznoj slici određuju se prema formuli:

$$g(x', y') = f(sx', sy')$$

. . . . . .

Projekat 3 - ekvalizacija histograma slike

Napisati program kojim se vrši ekvalizacija histograma ulazne slike.

. . . . . .

Histogram ekvalizacija predstavlja metod poboljšanja slike transformacijom intenziteta i može se opisati sledećim setom koraka:

Korak 1 : formiranje histograma slike

Histogram slike može se predstaviti nizom koji sadrži onoliko članova koliko različitih intenziteta piksela postoji u slici (pri izradi zadatka smatrati da su to intenziteti od 0-255), a vrednost svakog pojedinačnog elementa niza predstavlja koliko piksela određenog intenziteta postoji u slici.

Korak 2 : transformacija intenziteta

Transformacija koja se vrši nad ulaznim pikselima sada se može opisati na sledeći način:

$$g_k = \frac{L_{max}}{M\,N} \sum_{i=0}^{k} n_i, \qquad k = 0, ..., L_{max}$$

L_{max} - maksimalna vrednost intenziteta piksela u slici

n_i - broj piksela u slici koji imaju intenzitet i

M, N - dimenzije slike

Dakle, svaki piksel inteziteta $s_k, \quad k = 0, ..., L_{max}$ u ulaznoj slici, zamenjuje se pikselom intenziteta g_k određenog na način koji je prethodno opisan.

Primer: Na slici 2 prikazan je jedan primer procesiranja slike primenom histogram ekvalizacije. Ukoliko niz H(s_k) predstavlja histogram slike, pri čemu $s_k = 0, ..., 255$ predstavlja indeks niza H, i ujedno odgovarajući intenzitet piksela, a vrednost ovog elementa niza predstavlja broj piksela u ulaznoj slici sa intenzitetom s_k, tada za ulaznu sliku važi:

H(38)=16

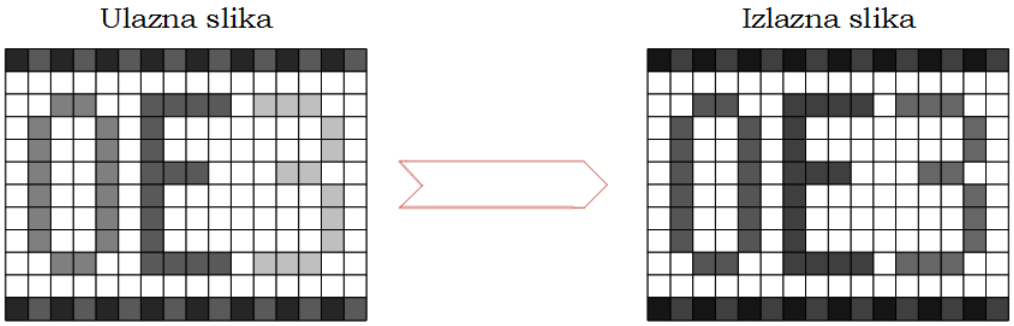
H(89)=32

H(127)=16

H(191)=13

H(255)=117

dok su svi ostali elementi niza jednaki 0. Dimenzije ulazne slike su 12×16.



Slika 2: Primer histogram ekvalizacije slike

Na osnovu prethodno opisanog algoritma važi da je:

$$g_{38} = \frac{255}{192} \sum_{i=0}^{38} n_i = \frac{255}{192} \cdot 16 = 21$$

$$g_{89} = \frac{255}{192}\sum_{i=0}^{89} n_i = \frac{255}{192}(16+32) = 63$$

$$g_{127} = \frac{255}{192}\sum_{i=0}^{127} n_i = \frac{255}{192}(16+32+16) = 85$$

$$g_{191} = \frac{255}{192}\sum_{i=0}^{191} n_i = \frac{255}{192}(16+32+16+13) = 102$$

$$g_{255} = \frac{255}{192}\sum_{i=0}^{255} n_i = \frac{255}{192}(16+32+16+13+117) = 255$$

Odnosno, piksel sa intenzitetom 38 slika se u piksel intenziteta 21, i analogno za sve ostale piksele.

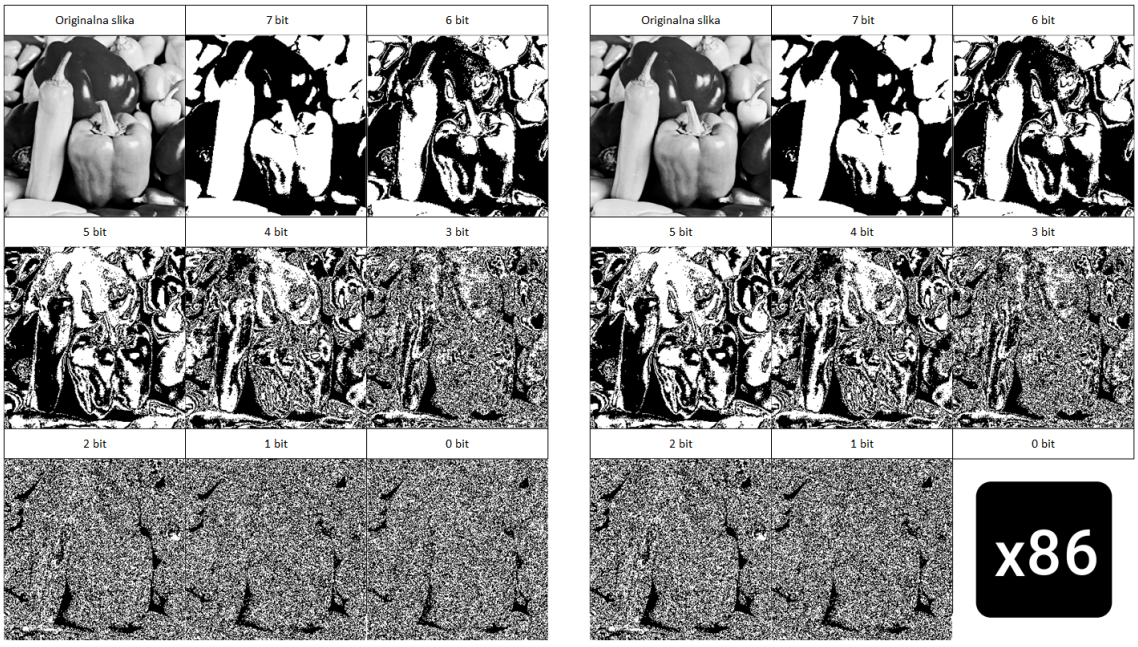
. . . . . .

Projekat 4 - otkrivanje/skrivanje tajne poruke

Napisati program kojim se vrši otkrivanje tajne poruke koja se nalazi u slici i zapisuje je u novi PGMA file, a zatim u ulaznu sliku sakriva proizvoljnu novu poruku.

. . . . . .

U slikama koje su zapisane u 8-bitnom formatu, mogu se izdvojiti pojedine bitske ravni, kao što je to prikazano na slici 3a. Pokazuje se da na izgled slike dominantno utiče nekoliko najviših bitskih ravni, dok najniže ravni jako malo, moglo bi se reći i neprimetno, utiču na sam izgled slike. Zbog ove osobine, u poslednju bitsku ravan moguće je sakriti tajnu poruku koja se posmatranjem cele slike ne može primetiti, ali se može dobiti izdvajanjem najniže bitske ravni iz slike, kao što se vidi na slici 3b.



(a) originalna slika (b) slika sa skrivenom porukom

Slika 3: Izgled bitskih ravni slike

Ukoliko tajna poruka/slika sadrži više nivoa sivog (nije crno-bela), njena bitska ravan najveće težine umeće se kao bitska ravan najmanje težine u sliku u koju se sakriva poruka. Ukoliko je slika crno-bela sasvim je svejedno koja se bitska ravan tajne poruke izdvaja, jer su sve identične. Nakon izdvajanja najniže bitske ravni, a pre zapisivanja poruke u izlazni PGMA file, potrebno je sve elemente pomnožiti sa 255 kako bi se ostvario što bolji kontrast.

Projekat 5 - set jednostavnih akcija nad slikom

Napisati program koji sa učitava sliku u PGMA formatu i nakon učitavanja slike, sa standardnog ulaza učitava se akcija koju je potrebno izvršiti nad slikom.. Naziv slike koja se učitava unosi se sa standardnog ulaza. Akcija koja se izvršava nad slikom definiše se unošenjem nekog od sledećih karaktera:

rr - rotiranje slike u desno

rl - rotiranje slike u levo

rf - rotiranje slike za 180 stepeni

mh - ogledanje slike oko leve ivice slike

mv - ogledanje slike oko desne ivice slike

Rezultat izvršavanja programa je PGMA file koji sadrži sliku nad kojom je obavljena željena akcija.

Projekat 6 - binarizacija slike iterativnim određivanjem praga binarizacije

Napisati program koji učitava sliku u PGMA formatu, zatim primenom iterativnog metoda određuje prag binarizacije T i zatim vrši binarizaciju slike tako što se svi pikseli sa vrednošću većom od T postavljaju na 255, a svi manji od T na 0.

Iterativno određivanje praga zasniva se na formiranju histograma slike (više informacija o histogramu može se pročitati u tekstu projekta 3) i definisanju početnog praga binarizacije T. Zatim se računaju srednje vrednosti osvetljaja piksela intenziteta manjeg od T ($\mathbf{T}_1$) i piksela intenziteta većeg od T ($\mathbf{T}_2$). Novi prag binarizacije dobija se kao srednja vrednost pragova T_1 i T_2. Ovaj postupak ponavlja se dok god je razlika između pragova dobijenih u dva susedna koraka veća od neke predefinisane konstante.

Rezultat izvršavanja programa je slika u PGMA formatu binarizovana primenom dobijenog praga.

Šifrovanje i dešifrovanje

Projekat 7 - (de)šifrovanje primenom Rail Fence algoritma

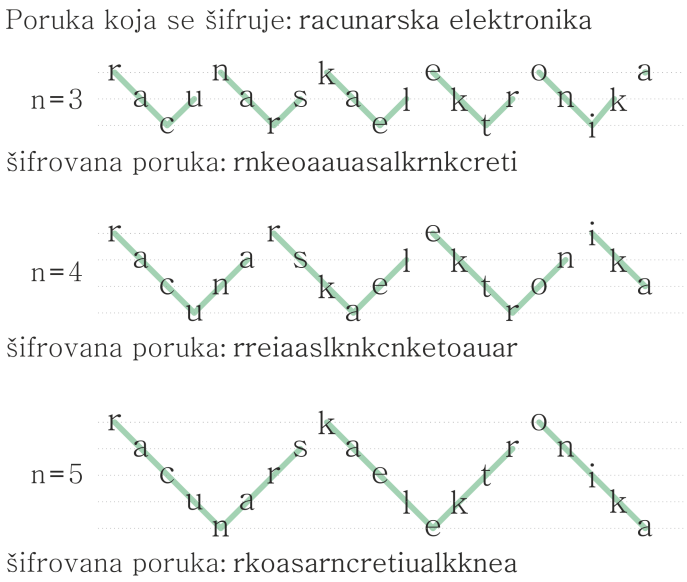
Napisati program koji učitava ulazni fajl koji u prvom redu sadrži

mode $\in \{e, d\}$ - karakter koji definiše da li tekst koji se nalazi u fajlu treba šifrovati (e) ili dešifrovati (d) primenom Rail Fence algoritma.

n - parametar koji definiše broj redova koji se koristi ili je korišćen prilikom enkripcije fajla

koji su razdvojeni jednim znakom razmaka, a od drugog reda na dalje nalazi se tekst koji je potrebno obraditi. Kao rezultat izvršavanja programa, potrebno je u izlazni fajl, prema formatu specificaranom za ulazni fajl upisati mode i broj redova, kao i rezultat obrade ulaznog teksta (parametar n je isti kao i u ulaznom fajlu, parametar mode je suprotan od onog koji je definisan u ulaznom fajlu).

Rail Fance algoritam zasniva se na zapisivanju poruke u cik-cak formatu u n redova, nakon čega se šifrovana poruka dobija čitanjem red po red. Na slici 4 je prikazan primer šifrovanja primenom Rail Fence algoritma za različite vrednosti parametra n.



Slika 4: Primer šifrovanja primenom Rail Fence algoritma

Dešifrovanje poruke koja je šifrovana primenom Rail Fance algoritma, trivijalna je ukoliko je poznat broj redova u kojem se vrši šifrovanje. Ukoliko pogledamo sam algoritam, može se primetiti postojanje ciklusa, pri čemu jedan ciklus predstavlja skup karaktera koji se nalaze između dva karaktera u prvom redu. Na slici 4 karakteri koji pripadaju jednom ciklusu spojeni su punom linijom. Broj karaktera u jednom punom ciklusu je $2n - 2$ i ukoliko postoji k punih ciklusa, broj karaktera u i-tom redu (i = 1..n) dat je izrazom:

$$\text{num\_of\_c}(i) = \begin{cases} k, & i \in \{1, n\} \\ 2k, & 1 < i < n \end{cases}$$

Broj punih ciklusa se prilikom dešifrovanja dobija deljenjem ukupnog broja karaktera u poruci sa dužinom ciklusa. Ukoliko postoji ostatak pri deljenju, jednostavno se određuje koliko

još karaktera postoji u kom redu. Nakon određivanja broja karaktera u redovima, moguće je podeliti šifrovanu poruku u redove i zatim originalnu poruku dobiti cik-cak iščitavanjem iz redova.

.....

Projekat 8 - (de)šifrovanje primenom Row Transposition algoritma

Napisati program koji učitava ulazni fajl koji u prvom redu sadrži

mode $\in \{e, d\}$ - karakter koji definiše da li tekst koji se nalazi u fajlu treba šifrovati (e) ili dešifrovati (d) primenom Row Transposition algoritma.

n - broj koji definiše koliko puta se prilikom šifrovanja ponavlja algoritam

koji su razdvojeni jednim znakom razmaka, u drugom redu sadrži ključ koji ima maksimalno 9 cifara i specificira redosled iščitavanja kolona prilikom formiranja šifrovane poruke, a od trećeg reda na dalje nalazi se tekst koji je potrebno obraditi. Kao rezultat izvršavanja programa, potrebno je u izlazni fajl, prema formatu specificaranom za ulazni fajl upisati mode, parametar n i ključ, kao i rezultat obrade ulaznog teksta (parametar n i ključ su isti kao i u ulaznom fajlu, parametar mode je suprotan od onog koji je definisan u ulaznom fajlu).

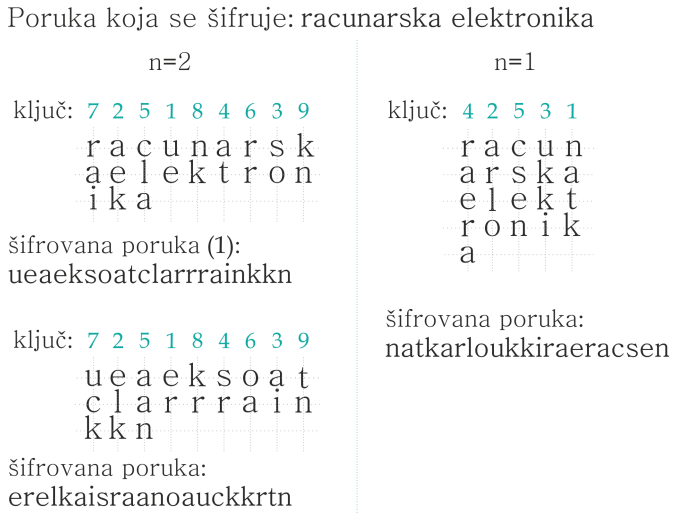
.....

Row Transposition algoritam može se podeliti u dva osnovna koraka:

Korak 1: Podela poruka u k redova dužine ključa, pri čemu je broj k zavisan od dužine poruke koja se šifruje

Korak 2: Iščitavanje poruka po kolonama prema redosledu definisanim ključem.

Ovaj algoritam ponavlja se n puta, nakon čega se dobija šifrovana poruka. Na slici 5 je prikazan primer šifrovanja poruke primenom Row Transposition algoritma.



Slika 5: Primer šifrovanja primenom Row Transposition algoritma

Dešifrovanje poruke koja je šifrovana primenom Row Transposition algoritma, trivijalna je ukoliko je poznat broj ponavljanja algoritma i ključ kojim se vrši šifrovanje. Broj karaktera po kolonama dobija se podelom broja karaktera u poruci sa dužinom ključa, nakon čega se može izvršiti podela šifrovanog teksta po kolonama, a zatim se išitavaju karakteri ciklično po kolonama u redosledu definisanom ključem.

Projekat 9 - (de)šifrovanje primenom Vigenèr-ovog/Beaufort-ovog algoritma

Napisati program koji učitava ulazni fajl koji u prvom redu sadrži

mode $\in \{e, d\}$ - karakter koji definiše da li tekst koji se nalazi u fajlu treba šifrovati (e) ili dešifrovati (d)

type $\in \{v, b\}$ - karakter koji definiše da li se (de)šifrovanje obavlja primenom Vigenèr-ovog (v) ili Beaufort-ovog (b) algoritma

key_type $\in \{s, a\}$ - karakter koji definiše da li (de)šifrovanje obavlja korišćenjem standardne (s) ili autokey metode (a)

koji su razdvojeni jednim znakom razmaka, u drugom redu nalazi se reč koja predstavlja ključ koji se koristi prilikom formiranja šifrovane poruke, a od trećeg reda na dalje nalazi se tekst koji je potrebno obraditi. Kao rezultat izvršavanja programa, potrebno je u izlazni fajl, prema formatu specificaranom za ulazni fajl upisati mode, type, key_type i ključ, kao i rezultat obrade ulaznog teksta (parametari type, key_type i ključ su isti kao i u ulaznom fajlu, parametar mode je suprotan od onog koji je definisan u ulaznom fajlu).

Na slici 6 prikazana je matrica koja se koristi prilikom upotrebe ovih algoritama.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
C	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
D	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
E	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
F	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
G	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
H	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
I	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
J	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
K	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
L	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
M	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
P	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
Q	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
R	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
S	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
T	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
U	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
V	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
W	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
X	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
Y	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
Z	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A

Slika 6: Matrica koja se koristi prilikom šifrovanja

Naime, ova dva algoritma su jako slična, samo se razlikuju delovi matrice koji se koriste kao ključ:

Vigenèr-ovo algoritam matricu sa slike 6 interpretira na sledeći način:

Prva kolona predstavlja slova ključa, prvi red predstavlja slova poruke koja se šifruje, a u preseku reda koji odgovara slovu ključa i kolone koja odgovara slovu poruke nalazi se slovo šifrovane poruke. Ovo se takođe može predstaviti formulom:

$$C_i = (M_i + K_i) \bmod 26$$

Dešifrovanje poruke šifrovane ovim algoritmom može se predstaviti formulom:

$$M_i = (C_i - K_i) \bmod 26$$

Beaufort-ov algoritam matricu sa slike 6 interpretira na sledeći način:

Prvi red predstavlja slova poruke koja se šifruju, a unutrašnja matrica predstavlja slova ključa. Slovo šifrovane poruke dobija se tako što se u koloni koja odgovara slovu poruke nađe slovo ključa, i slovo iz prve kolone kome odgovara taj red predstavlja slovo šifrovane poruke. Ovo se takođe može predstaviti formulom:

$$C_i = (K_i - M_i) \bmod 26$$

Dešifrovanje poruke šifrovane ovim algoritmom može se predstaviti formulom:

$$M_i = (K_i - C_i) \bmod 26$$

gde je M_i slovo poruke koja se šifruje, K_i slovo ključa i C_i slovo dobijeno šifrovanjem.

U slučaju da je poruka koja se šifruje kraća od dužine ključa, koristi se samo potreban broj slova iz ključa, nezavisno da li se koristi standardna ili autokey metoda. Ukoliko je poruka koja se šifruje duža od ključa, način produžavana ključa razlikuje se u slučaju ove dve metode:

Standardna metoda zasniva se na periodičnom ponavljanju ključa potreban broj puta, tako da se pokriju sva slova poruke

Autokey metoda zasniva se na dodavanju slova iz poruke koja se šifruje na originalni ključ, dok se ne pokriju sva slova poruke

.....

Projekat 10 - (de)šifrovanje upotrebom Trifid algoritma

Napisati program koji učitava ulazni fajl koji u prvom redu sadrži karakter **mode** $in\{e, d\}$ koji definiše da li tekst koji se nalaziu fajlu treba šiforvati (e) ili dešifrovati (d) primenom Trifid algoritma. Od drugog reda na dalje nalzi se tekst koji je potrebno obraditi. Kao rezultat izvršavanja programa, potrebno je u izlazni fajl, prema formatu specificiranom za ulazni fajl upisati mode, koji je suprotan od onog koji je definisan u ulaznom fajlu i rezultat obrade.

.....

Ovaj algoritam prilikom šifrovanja i dešifrovanja koristi 3 matrice karaktera dimenzija 3×3:

```
Layer 1        Layer 2        Layer 3
  1 2 3          1 2 3          1 2 3
1 A B C        1 J K L        1 S T U
2 D E F        2 M N O        2 V W X
3 G H I        3 P Q R        3 Y Z .
```

Postupak šifrovanja najlakše je objasniti na primeru:

```
Poruka koja se šifruje:   r a c u n a r s k a   e l e k t r o n i k a .
Layer:                    2 1 1 3 2 1 2 3 2 1   1 2 1 2 3 2 2 2 1 2 1 3
Kolona:                   3 1 3 3 2 1 3 1 2 1   2 3 2 2 2 3 3 2 3 2 1 3
Red:                      3 1 1 1 2 1 3 1 1 1   2 1 2 1 1 3 2 2 3 1 1 3
```

Sledeći korak je formiranje trojki iščitavanjem redova sleva na desno:

```
211 321 232 112 123 222 121 331 332 131 212 322 233 232 133 111 213 111 212 113
223 113
```

a zatim se ovde trojke koduju ponovo korišćenjem istih matrica, tako što prvi broj predstavlja layer, drugi kolonu, a treći red, tako da je dobijena šifrovana poruka: `jtodhnbuxcmwroiapamgug`.

.....

Pravimo igrice

Projekat 11∗ - Igra memorije

Potrebno je realizovati igru memorije (`http://www.primarygames.com/puzzles/match_up/memory/`). Primer izgleda prozora prikazan je na slici 7. Tabla se sastoji od matrice karata dimenzija M × N. Broj različitih karata na tabli je $\frac{M \times N}{2}$. Cilj igre je pogađanje parova karata. Igra funkcioniše tako što igrač u svakom potezu može da otvori dve karte, i ukoliko su one iste, one ostaju otvorene i igrač dobija **K** poena. U slučaju da karte nisu iste igrač gubi $\frac{\mathbf{K}}{4}$ poena i karte se zatvaraju. K je broj po izboru veći od 0, deljiv sa 4. Igrač bira kartu koju želi da otvori korišćenjem tastera tastature za pomeranje u 4 smera, i korišćenjem dodatnog tastera za potvrdu otvaranja karte. Karta koja bi se otvorila pritiskom tastera za potvrdu otvaranja karte, treba da bude uokvirena. Potrebno je da se u svakom trenutku na ekranu nalazi trenutni broj poena koje je igrač osvojio. Treba obratiti pažnju da nije moguće ponovo otvoriti već otvorenu kartu.

Igrica se završava pogađanjem svih karata ili pritiskom tastera ESC.

Slika 7: Igra memorije

. . . . . .

Projekat 12∗ - Pattern memory

Potrebno je realizovati igru *Pattern memory* (`http://www.primarygames.com/puzzles/memory/patternmemory/`). Primer izgleda prozora u trenutku prikaza pattern-a i u trenutku kada igrač treba da pogađa prethodno prikazani pattern prikazan je na slici 8. Tabla se sastoji iz matrice dimenzija N×N, pri čemu su minimalne dimenzije 3×3. Potrebno je podržati minimum 5 različitih pattern-a (5 nivoa). Igrica se sastoji u tome da se najpre u određenom vremenskom intervalu (t_{show}) na ekranu prikazuje zadati pattern, koji se nakon isteka tog vremena uklanja nakon čega korisnik pogađa zadati pattern. Pogađanje pattern-a korisnik obavlja tako što se kroz matricu kreće pomoću 4 tastera na tastaturi, i koristi dodatni taster za potvrdu polja za koje misli da pripada pattern-u. Polje koje bi se trenutno selektovalo pritiskom tastera za potvrdu, treba da bude obojeno nekom bojom koja se razlikuje od boje kojom se prikazuje pattern, a koje se nakon pritiska tastera za potrvdu boji u boju koja se koristi za prikazivanje pattern-a. Nakon što je koristik K puta, pritiskom tastera za potvrdu izabrao

željena polja, proverava se uneti pattern. Ukoliko je pattern koji je korisnik uneo, jednak onom koji je inicijalno prikazan, korisnik dobija poene, i prelazi se na sledeći nivo, dok se u suprotnom završava igra. Broj K opciono može da varira u zavisnosti od nivoa i predstavlja broj elemenata u prikazanom pattern-u. t_{show} je proizvoljan razuman vremenski interval koji takođe opciono može varirati između nivoa. Broj N se definiše unapred i isti je za svaki nivo. Takođe, treba obratiti pažnju da kada je korisnik označio određeno polje, ukoliko pokuša da ga u istom nivou označi ponovo, ne treba ništa raditi.

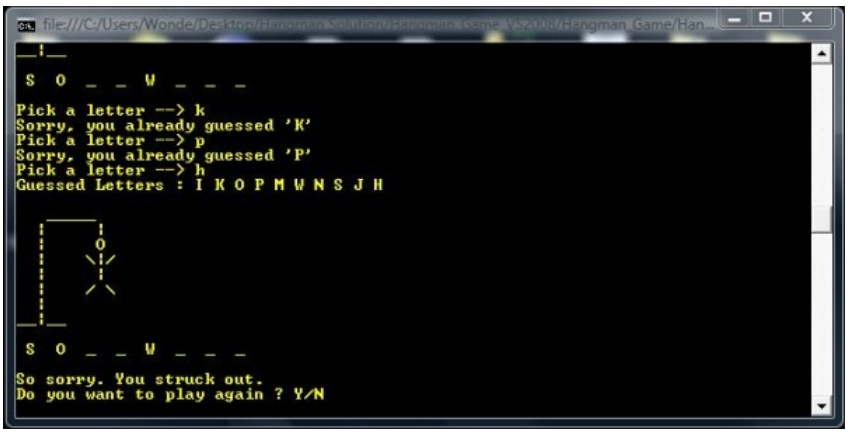
Slika 8: Igrica Pattern memory

. . . . . .

Projekat 13 - Vešala

Potrebno je realizovati igru Vešala (*Hangman*). Pogledati `http://www.playhangman.com/PH.asp?g=car`. Primer izgleda prozora prikazan je na slici 9. Potrebno je uneti niz pojmova koji se pogađaju u igrici (minimum 10), i omogućiti igraču da unošenjem slova sa standardnog ulaza pogađa pojam. Ukoliko slovo koje je uneto sa standardnog ulaza ne postoji, dodaje se deo Čiča Gliše, dok ukoliko postoji, upisuje se na odgovarajuću poziciju u reči. Ukoliko je slovo već ranije pogađano, potrebno je o tome obavestiti korisnika, i ne preduzimati nikakve druge akcije. Svaki put kada se sa standardnog ulaza unese novo slovo, potrebno ga je ispisati na ekranu. Kada korisnik pogodi trenutni pojam, omogućava mu se da pogađa novi. Ukoliko korisnik pogreši slova dovoljan broj puta, tako da se iscrta ceo Čiča Gliša, gubi igru.

Igricu je moguće prekinuti i pritiskom tastera ESC.

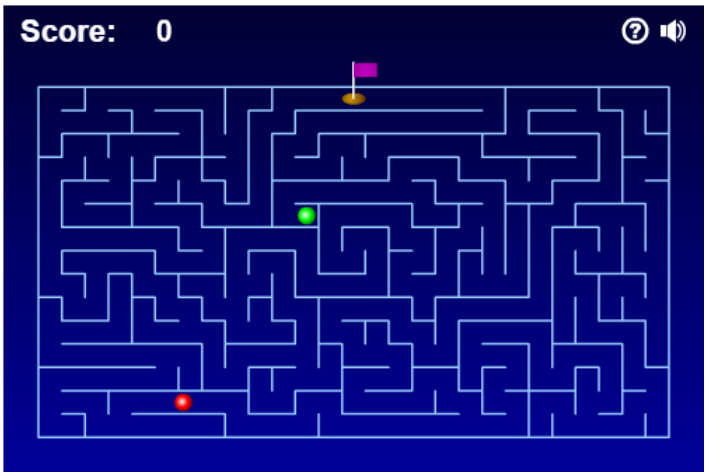


Slika 9: Igrica vešala

. . . . . .

Projekat 14∗ - Maze race

Potrebno je realizovati igru *Maze race*:
`http://www.mobogenie.com/download-maze-race-king-2645086.html`
Primer izgleda prozora prikazan je na slici 10. U lavirintu treba da postoje dva kvadratića jedan zeleni i jedan crveni koji su sastavljeni od dva blok karaktera (DBh). Potrebno je omogućiti kretanje zelenog kvadratića kroz lavirint koristeći tastere na tastaturi. Crveni kvadradić kreće se prema cilju prema unapred definisanoj putanji (bez upadanja u zamke) nekom konstantnom brzinom koja je manja od brzine kretanja zelenog kvadratića. Svaki put kada se crveni kvadratić pomeri bliže cilju, igrač gubi po 1 poen od maksimalnog broja poena koji osvaja ukoliko stigne do cilja pre crvenog kvadratića. Ukoliko crveni kvadratić stigne do cilja pre igrača, igrač gubi igru. Cilj, koji je na slici 10 označen zastavicom, označiti kvadratićem napravljenim od 2 karaktera sa ASCII kodom B2h, u boji po izboru. Za iscrtavanje zidova lavirtinta, koristiti karaktere po izboru koji pripadaju proširenom skupu ASCII kodova (`http://www.asciitable.com/`).

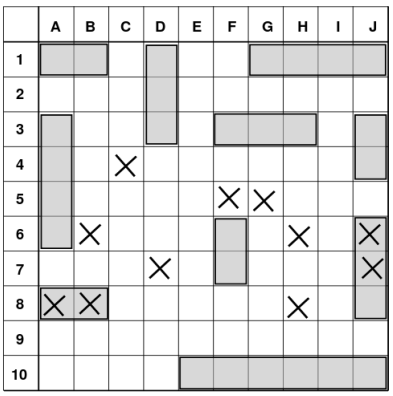


Slika 10: Igrica maze race

. . . . . .

Projekat 15 - Battleship

Potrebno je realizovati igru potapanje brodića (*Battleship*). Standardni izgled table za brodiće prikazan je na slici 11.



Slika 11: Igrica battleship

. . . . . .

Potrebno je u aplikaciji najpre omogućiti igračima da unesu imena fajlova u kojima se nalaze njihove mape rasporeda brodića a koji treba da sadrže 10 redova sa po 10 karaktera razdvojenih razmakom u sledećem formatu:

```
1 1 - 2 - - 3 3 3 3
- - - 2 - - - - - -
6 - - 2 - 4 4 4 - 5
6 - - - - - - - - 5
6 - - - - - - - - -
6 - - - - 9 - - - 0
- - - - - 9 - - - 0
7 7 - - - - - - - 0
- - - - - - - - - -
- - - - 8 8 8 8 8 8
```

Ovaj fajl bi predstavljao opis mape kao na slici 11. Mapa sadrži 10 brodića numerisanih ciframa 0-9, i to 1 brodić od 5 polja, 2 brodića od 4 polja, 3 brodića od 3 polja i 4 brodića od 2 polja, koji mogu biti postavljeni horizontalno ili vertikalno. Nakon što igrači unesu imena fajlova u kojima se nalaze njihove mape, potrebno je iscrtati dve mape koje odgovaraju svakom pojedinačnom igraču:

```
     Player 1                    Player 2
   A B C D E F G H I           A B C D E F G H I
0  - - - - - - - - -        0  - - - - - - - - -
1  - - - - - - - - -        1  - - - - - - - - -
2  - - - - - - - - -        2  - - - - - - - - -
3  - - - - - - - - -        3  - - - - - - - - -
4  - - - - - - - - -        4  - - - - - - - - -
5  - - - - - - - - -        5  - - - - - - - - -
6  - - - - - - - - -        6  - - - - - - - - -
7  - - - - - - - - -        7  - - - - - - - - -
8  - - - - - - - - -        8  - - - - - - - - -
9  - - - - - - - - -        9  - - - - - - - - -
```

Borba se odvija tako što igrači sa standardnog ulaza unose koordinate polja koje žele da gadjaju u formi slovo pa broj, koji nisu razdvojeni razmakom (A5, F3, ...). Ukoliko je igrač promašio protivnički brod, potrebno je na tom polju upisati x i omogućiti drugom igraču da pogađa, dok u slučaju da je igrač pogodio protivnički brod, na tom polju potrebno je iscrtati blok karakter (DBh) i omogućiti istom igraču da pogadja ponovo. Pobednik je onaj igrač koji prvi pogodi sve protivničke brodiće

.....

Projekat 16∗ - Snake

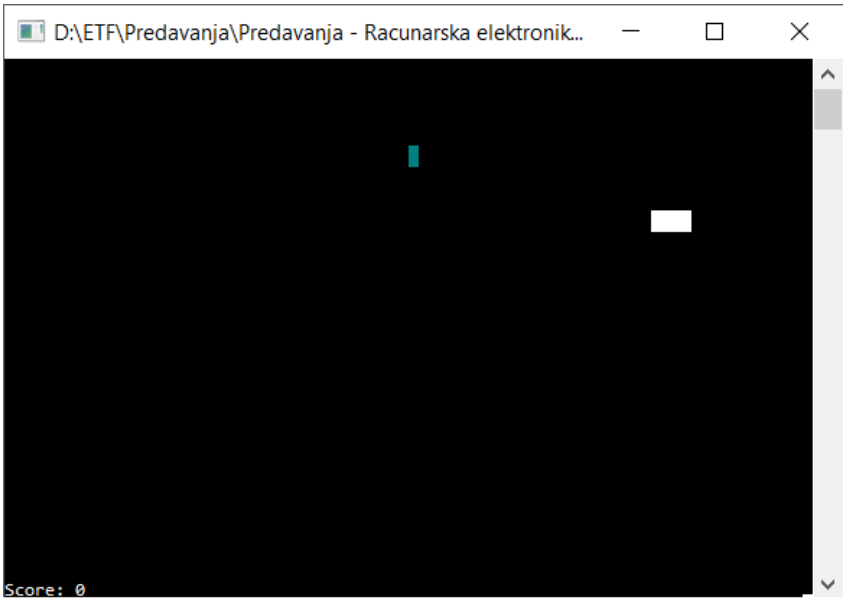
Potrebno je realizovati igru zmijica (*Snake*).

Koristeći kursor tastere na tastaturi pomerati zmijicu u četiri pravca. Blokove zmijice realizovati u vidu praznog prostora (*spacebar* – FFFFh), tako da je prazan prostor kojim se iscrtava zmijica obojen u boju drugačiju od pozadine. Hrana se pojavljuje na nasumično generisanim lokacijama (korišćenjem *random* funkcije) i takođe se realizuje korišćenjem praznog prostora obojenog u boju drugačiju od zmijice i pozadine.

Početna pozicija zmijice je na sredini igrališta i početna dužina zmijice je četiri polja. Igrica se napušta pritiskom na taster ESC, prilikom kolizije sa preprekom ili prilikom kolizije zmijice sa samom sobom. Nakon završetka igre, ispisuje se broj osvojenih poena.

Prilikom pokretanja programa, i nakon napuštanja igre, program treba da uđe u glavni meni, u kojem je moguće odabrati početak igre, napuštanje programa i brzinu kretanja zmijice.

Na slici 12 je prikazan primer izgleda prozora prilikom igranja igrice.



Slika 12: Igrica Snake.

. . . . . .

Projekat 17∗ - Pong

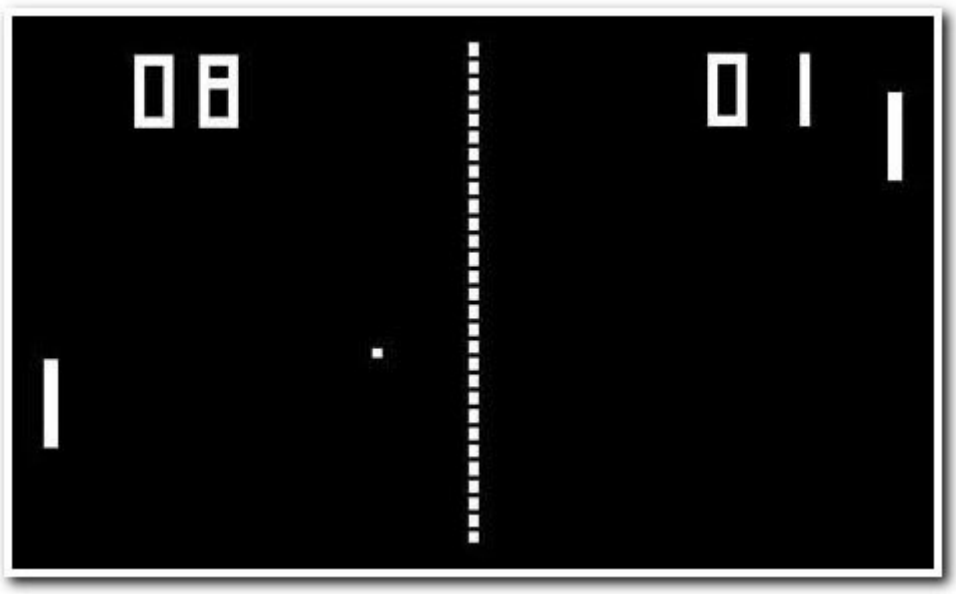
Napaviti igricu Pong za slučaj jednog i dva igrača (pogledati: `http://www.ponggame.org/`).

Igrači su predstavljeni pedalama veličine četiri prazna bloka. Kuglica je takođe oblika praznog bloka, i početna pozicija joj je na sredini ekrana. Svaki put kada kuglica udari u pedalu ili gornju ili donju ivicu prozora, ona se odbija pod istim uglom u odnosu na normalu pedale/ivice, samo u suprotom smeru. Takođe, svaki put kada udari pedalu, brzina kuglice se povećava. Ukoliko kuglica dotakne levu ili desnu ivicu prozora, onda se završava takuće „dodavanje" kuglice. Broj poena igrača koji je odigrao poslednji potez dodavanja se inkrementira za jedan i počinje nova partija.

Ukoliko se izabere igranje ponga u dva igrača, potrebno je aktivirati 4 kursora na tastaturi: po dva za pokretanje oba igrača. U slučaju da se izabere igranje sa jednim igračem, aktiviraju se dva tastera za pokretanje pedale. Drugog igrača je potrebno realizovati u programu tako da se kreće u smeru u kom se odbila kuglica od pedalu/ivicu prozora konstantnom brzinom.

Igrica se završava kada se odigra 10 partija ili pritiskom tastera ESC. Nakon završetka igrice se ispisuje pobednik sa osvojenim brojem bodova i ponovo se ulazi u glavni meni gde se može izabrati igranje u paru, sa jednim igračem ili izlazak iz igrice.

Na slici 13 je prikazan primer izgleda prozora prilikom igranja igrice.



Slika 13: Igrica Pong.

. . . . . .

Projekat 18∗ - Asteroids

Potrebno je realizovati igricu Asteroids (pogledati: `http://www.freeasteroids.org/`).

Na početku igrice na sredini ekrana se nalazi svemirski brod veličine 4 prazna bloka. Svake sekunde se pojavljuje po jedan asteroid sastavljen od 8 praznih blokova na proizvoljnim pozicijama. Svemirski brod pritiskom tastera SPACE ispaljuje mine oblika jednog praznog bloka. Mina se kreće unapred zadatom brzinom, a ako se i svemirski brod kreće onda je brzina mine jednaka:

$$v_M = v_{M,const} + 3\, v_B,$$

gde je v_M brzina mine, $v_{M,const}$ unapred zadata konstantna brzina mine i v_B brzina broda.

Brod može da se drugačije orijentiše pritiskom tastera levo i desno na tastaturi gde se svakim pristkom obće u smeru kazaljke na satu (odnosno obrnutom smeru od kazaljke na satu) za približno 10°. Pritiskom kursora na gore se brod ubrzava tako što se doda inkrement na njegovu brzinu, a taster na dole ga usporava. U toku kretanja u svemiru postoji i trenje koje se ogleda u tome što se na svakih nekoliko polja smanjuje brzina broda koji staje ukoliko nije pritisnut taster na gore (pogledati video: `Asteroids.mp4`).

Kada mina udari u asteroid, asteroid biva raznet i nestaje, a broj bodova osvojen u igrici se uvećava za 100. Ukoliko svemirski brod udari u asteroid ili se novi asteroid pojavi na svemirskom brodu, onda se igrica završava i ispisuje se ukupan broj osvojenih bodova. Igrica se završava i pritiskom tastera ESC.

. . . . . .

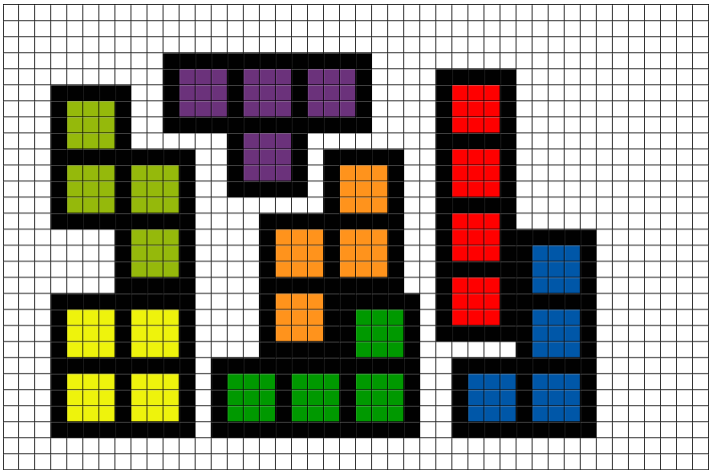
Projekat 19∗ - Tetris

Potrebno je napraviti igricu Tetris (pogledati: `http://www.freetetris.org/game.php`, `http://www.classicgamesarcade.com/game/21602/tetris-flash-arcade-game.html` i `https://www.youtube.com/watch?v=O0gAgQQHFcQ`).

U igrici je raspoloživo 6 komponenata sa slike 14 od kojih se jedna nasumično izabrana pojavljuje na sredini gornje ivice ekrana. Nova komponenta se pojavljuje na početku igrice i svaki

put kada prethodna komponenta dodirne donju ivicu prozora ili već naslagane komponente. Komponenta se pomera ka donjoj ivici prozora unapred izabranom konstantnom brzinom i može da se ubrza kursorom na dole na tastaturi. Kursori levo i desno pomeraju komponentu levo odnosno desno, a kursor ka gore je rotira za 90° u suprotnom smeru kazaljke na satu. Komponenta koja treba da se sledeća pojavi se crta levo od prozora u kome se slažu komponente. Sa desne strane se ispisuje broj bodova osvojenih u toku igranja igrice.

Svaki put kada se jedan red u potpunosti popuni, on se briše, a broj osvojenih poena se poveća za 100. Igrica se napušta kada komponenta dodirne gornju ivicu prozora ili pritisnom tastera ESC. Prilikom izlaska iz igrice ispisuje se ukupan osvojen broj poena.



Slika 14: Delovi u igrici tetris.

Na slici 15 je prikazan primer izgleda prozora prilikom igranja igrice.

Slika 15: Igrica Tetris.

. . . . . .

Projekat 20 - Tic Tac Toe

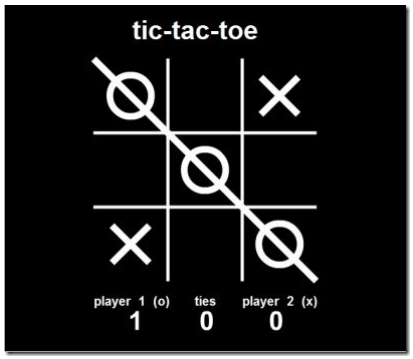
Potrebno je napraviti igricu Tic Tac Toe (pogledati: `http://www.classicgamesarcade.com/game/21622/tic-tac-toe.html`) za igranje u dve osobe.

U igrici se 9 praznih polja popunjava tako što se pritiska jedan od 9 tastera rezervisanih za igranje igrice, čime se upisuje u dato polje `X` ili `O`. Po pravilu prvom igraču odgovara `O`, a

drugom `X`, a igrači igraju naizmenično. Igra se završava kada jedan od igrača sakupi 3 ista simbola (`O` ili `X`) u vrsti, koloni ili dijagonalno. Tada se ispisuje da je dati igrač pobedio. Igrica može da se prekine i pritiskom tastera ESC.

Prilikom realizacije programa voditi računa o izgledu glavnog prozora igrice.

Na slici 16 je prikazan primer izgleda prozora prilikom igranja igrice.



Slika 16: Igrica Tic Tac Toe.

. . . . . .

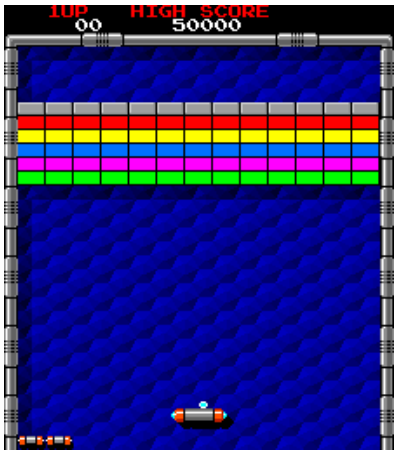
Projekat 21∗ - Arkanoid

Napraviti igricu Akranoid (pogledati: `http://www.classicgamesarcade.com/game/21603/arkanoid.html`).

Igrač u igrici kontroliše pedalu pomerajući je levo i desno tako da kuglica ne dodirne donju ivicu prozora. Kuglica i pedala se na početku nalaze na sredini donje ivice prozora i kuglica započinje svoje „kretanje" uvek pod istim uglom. Kada kuglica udari u neku od gornjih prepreka, ruši ih i igrač osvaja bodove. Udarom o bilo koju prepreku, izlazni ugao kuglice je jednak ulaznom u odnosu na normalu prepreke, pedale ili leve i desne ivice prozora. Prepreke u zavisnosti od boje nose različit broj poena: zelena - 80 poena, crvena - 90 poena, žuta - 120 poena, ljubičasta - 100 poena i plava - 50 poena. Siva prepreka se uništava ukoliko je dva puta udarena, a ako pedala uhvati srušenu sivu prepreku, povećava se dva puta.

Igrica se završava kada se unište sve prepreke ili kada se izgubi 10 života kuglice. Na ekranu je sve vreme potrebno ispisivati preostali broj života kuglice i broj osvojenih poena. Igrica može da se prekine i pritiskom tastera ESC.

Na slici 17 je prikazan primer izgleda prozora prilikom igranja igrice.



Slika 17: Igrica Arkanoid.

. . . . . .

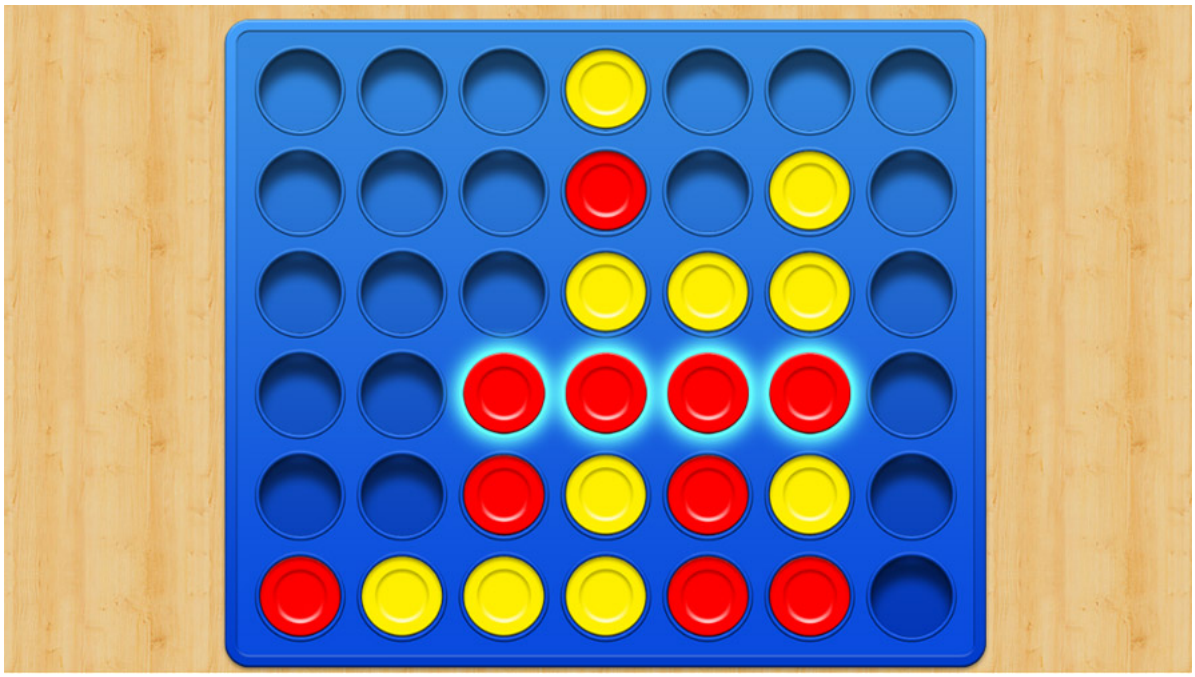
Projekat 22 - Connect4

Napraviti igricu Connect 4 (pogledati: `http://www.classicgamesarcade.com/game/21660/connect4.html`) za igranje u 2 osobe.

Tabla veličine 7×6 polja se sastoji od 42 kvadratna prazna polja (umesto kuglica). Igrači igraju naizmenično ubacujući kuglice (u ovom slučaju popunjavajući jedan kvadrat od 4 bloka istim blokom samo neke druge boje) od gore u kolonu. Ubacivanje se može realizovati korišćenjem 7 tastera na tastaturi, svaki za po jedan red. Trenutni igrač pritiskom odgovarajućeg tastera ubacuje jednu „kuglicu" u tekuću kolonu. Kolone trebaju da imaju labele sa oznakom tastera koji treba pritisnuti, a igrači treba da imaju legende sa bojom njihovih „kuglica".

Pobednik je igrač koji prvi sastavi 4 „kuglice" u koloni, vrsti ili dijagonalno. Ako se popune sva polja, a niko nema 4 kuglice u koloni/vrsti ili dijagonalno, onda je partija nerešena. Nakon završetka partijese ispisuje ko je pobednik, ondosno da je nerešena partija. Igrica može da se prekine i pritiskom tastera ESC.

Na slici 18 je prikazan primer izgleda prozora prilikom igranja igrice.



Slika 18: Igrica Connect 4.

. . . . . .

Projekat 23∗ - Pocket Tanks

Napraviti igricu Pocket Tanks (pogledati: `http://www.miniclipgamez.com/fullsizegame/669/play-pocket-tanks-game.html`) za dva igrača koji naizmenično igraju.

Igricu je potrebno realizovati tako da se između tenkova nalazi trougaona prepreka. Tenkovi su kvadrati sačinjeni od 4 bloka. Levi tenk može da se kreće samo ka desno, a desni na levo kursorima desno i levo na tastaturi. Položaj glavnog topa tenka je na početku igrice pod uglom od 45° za levi tenk i 135° za desni tenk. Položaj topa može da se menja pritiskom kursora gore i dole na tastaturi. Tenk ispaljuje minu pritiskom tastera SPACE. Mina može da poruši prepreku ukoliko u nju udari ili da smanji broj života tenka za 1. Tenkovi imaju po 10 života. U gornjem levom, odnosno desnom, uglu je potrebno ispisati preostali broj života tenka i ugao pod kojim je postavljen top.

Igrica se završava kada jedan od tenkova izgubi sve živote. Po završetku se ispisuje pobednik. Igricu je moguće prekinuti i pritiskom tastera ESC.

Na slici 19 je prikazan primer izgleda prozora prilikom igranja igrice.

Slika 19: Igrica Pocket Tanks.

. . . . . .

Projekat 24∗ - Bejeweled

Napraviti igricu Bejeweled (`http://www.classicgamesarcade.com/game/21730/bejeweled.html`). Prozor je podeljen na 8×8 polja u kojima su poređani kvadrati u 4 različite boje: crvene, žute, plave i zelene. Na početku igrice se popunjavaju polja kvadratima nasumičnih boja. Kursorima na tastaturi se pomera kroz kvadrate, a tasterom SPACE se markiraju blokovi koji bi trebalo da zamene mesta. Mesta mogu da zamene samo susedni blokovi u vrsti ili koloni i to samo ukoliko s zamenom dobijaju u vrsti ili koloni bar tri kvadrata iste boje. Program treba da registruje i grupe veća od tri polja iste boje.

Svaki put kada se složi drupa od više od tri polja, data polja se brišu. Polja koja su iznad izbrisanih polja se pomeraju ka dole, a dobijena prazna polja na vrhu se popunjavaju poljima nasumične boje (crvene, žute, plave ili zelene).

Igrica traje 3 minuta. Svaki put kada se složi više od tri polja iste boje, broj osvojenih poena se povećava za $100\,n$, gde je n broj polja. Na kraju izvršavanja igrice se ispisuje broj osvojenih poena. Igrica se može prekinuti pritiskom tastera ESC.

Slika 20: Igrica Bejeweled.

. . . . . .

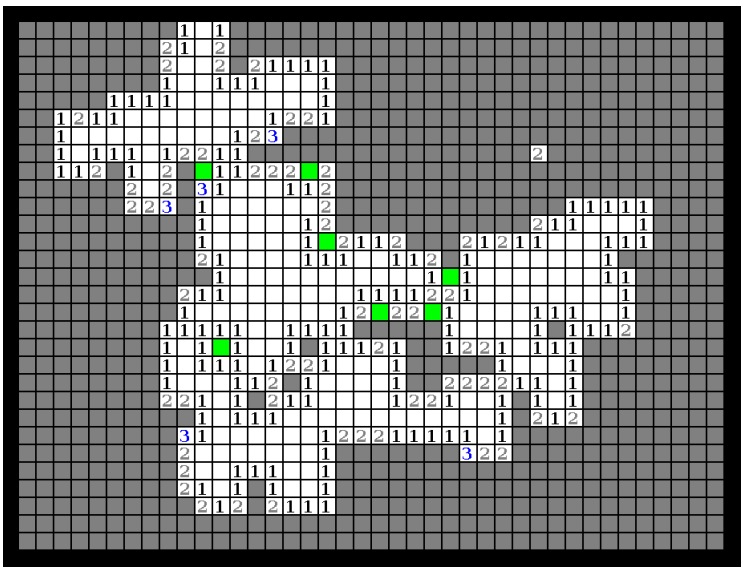
Projekat 25∗ - Minesweeper

Napraviti igricu Minesweeper:
`http://www.classicgamesarcade.com/game/21649/minesweeper.html`,
`https://en.wikipedia.org/wiki/Minesweeper_(video_game)`.

Prozor igrice (tabla) je podeljen na 9×9 polja koja su na početku igrice sva iste boje. Na 10 lokacija na tabli, odnosno iza 10 polja se nalaze mine koje su prikrivene. Kroz polja se kreće kursorima na tastaturi (na neki način označiti polje na kome se trenutno nalazi kursor). Polja napraviti veličine 2×2.

Pritiskom tastera SPACE se otvara polje. Ukoliko se na polju ne nalazi mina, rekurzivno se otvaraju i sva susedna polja (promeniti boju ovih polja) na kojima nema mine. Na ivicama otvorene oblasti bez mina se ispisuje broj koliko se mina nalazi pored datog polja u redu, koloni i djagonalno. Pritiskom na minu se završava igrica. Ako se pretpostavlja da se iza datog polja nalazi mina, onda se to polje označava pritiskom tastera Left Shift i tada dato polje postaje druge boje.

Igrica se završava kada se otvore sva polja na kojima nema mine i označe sva polja sa minama. Ukoliko se otvori polje na kome se nalazi mina, takođe se završava igrica. Pritiskom tastera ESC omogućiti prekid programa.



Slika 21: Igrica Minesweeper.

. . . . . .

Projekat 26∗ - 2048

Napraviti igricu 2048 (`https://en.wikipedia.org/wiki/2048_(video_game)`).

Igrica se igra na 4×4 mreži kao na slici 22. Na početku igrice na mreži se popunjava nasumično jedno polje kvadratom na kome je ispisan broj koji predstavlja stepen dvojke. Svi stepeni dvojke imaju odgovarajuću boju polja. Pritiskom kursora na tastaturi se popunjena polja pomeraju skroz levo, desno, gore ili dole.

Ukoliko se pomeranjem dodirnu dva polja sa istim stepenom dvojke, ona se zamenjuju poljem čija je vrednost jednaka zbiru ova dva polja, a ukupan broj poena se inkrementira za tu istu vrednost. U slučaju ako se desi da postoje 3 susedna polja koja imaju istu vrednost, onda se dva koja su bliža onoj granici mreže na koju ukazuje pritisnuti kursor zamenjuju jednim poljem sa duplo većom vrednošću.

Svaki put kada se pritisne neki od kursora na tastaturi, nakon pomeranja svih polja u smeru tog kursora, na neko od praznih polja se nasumično generiše novi element. Vrednost na polju

se generiše nasumično i to 2 sa verovatnoćom 5/8, 4 sa verovatnoćom 2/8 i 8 sa verovatnoćom 1/8.

Igrica se završava pritiskom tastera ESC ili kada na mreži ne postoji više nijedno prazno polje, a susedni elemenati ne mogu pomeranjem da daju jedno polje.

Slika 22: Igrica 2048.

Obrada zvuka

PCM format

PCM format je binarni format za zapis zvuka. Datoteka se sastoji iz odbiraka zvučnog signala kodovanog sa 16 bita. Ulazni signal se predstavlja kao 16-bitni označeni ceo broj, što znači da su mu odbirci u opsegu $[-2^{15}, 2^{15} - 1]$. Odbirci u PCM fajlu se čitaju tako što se uzme podatak od 16 bita, odnosno 2 bajta.

Veličina PCM fajla je jednaka broju odbiraka zvučnog signala pomnoženog sa 2B. To znači da fajl sa 5000 odbiraka zauzima 10000 B.

Pravljenje PCM fajla

Proizvoljna melodija u MP3 formatu može da se konvertuje u WAV format korišćenjem programa Audacity© koji se može skinuti sa linka: `http://www.audacityteam.org/`. Uputstvo za konverziju MP3 fajla u WAV fajl je dato na linku: `https://www.3cx.com/docs/converting-wav-file/`.

Za konverziju WAV fajla u PCM format koji se koristi prilikom izrade projekata koristi se skripta `convert_pcm.m` pisana za MATLAB.

. . . . . .

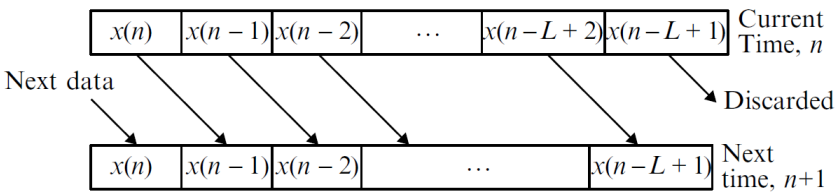
Projekat 27 - FIR filtar sa signalnim baferom

Potrebno je napraviti filtar niskih učestanosti. Projektovani filtar „propušta" niske učestanosti do frekvencije $f_g = 5$ kHz, dok na visokim učestanostima slabi signal 80 dB. Ulazni signal je dat u pcm formatu, a koeficijenti filtra su izračunati i nalaze se u fajlu `fir.txt`. Koeficijenti FIR filtra se mogu na početku programa kao konstantni niz.

Potrebno je napraviti program koji filtrira ulazni signal u pcm formatu korišćenjem FIR filtra čiji su koeficijenti dati u fajlu `fir.txt`. Inicijalizovati signalni bafer u koji će biti smešteno L odbiraka ulaznog signala. U svakom ciklusu obrade se čita po jedan odbirak ulaznog signala i vrši računanje signala na izlazu korišćenjem formule:

$$y[n] = \sum_{l=0}^{L} h[l]\, x[n-l],$$

gde je L broj odbiraka FIR filtra, $x[n]$ ulazni signal, a $h[n]$ odbirak FIR filtra. Nakon obrade jednog podatka se čita novi odbirak iz ulaznog fajla i vrši pomeranje ostalih odbiraka u signalnom baferu kao na slici 23.



Slika 23: Signalni bafer.

Nakon završetka čitanja ulaznog fajla, potrebno je generisati izlazni fajl u kome se nalaze odbirci izlaznog signala $y[n]$ u pcm formatu. Nazvati izlazni fajl kao ulazni `*_out.pcm`, gde je `*` ime ulaznog fajla (npr. ako je ulazni fajl `ime.pcm`, izlazni je `ime_out.pcm`).

Rezultat izvršavanja programa verifikovati na fajlu `input.pcm` tako što će se pokretanjem skripte u MATLAB-u `compare.m` prikazati rezultati filtriranja u MATLAB-u sa rezultatima filtriranja korišćenjem realizovanog programa. Dati dijagrami bi trebalo da se slažu.

Izmeriti vreme izvršavanja programa.

.....

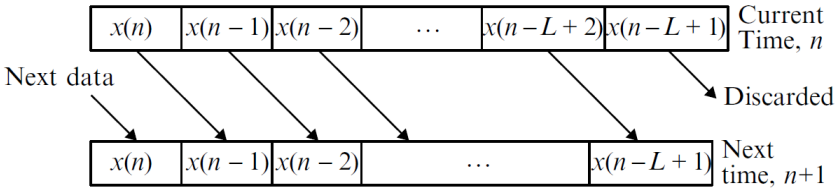
Projekat 28 - FIR filtar korišćenjem simetrije

Potrebno je napraviti filtar niskih učestanosti. Projektovani filtar „propušta" niske učestanosti do frekvencije $f_g = 5$ kHz, dok na visokim učestanostima slabi signal 80 dB. Ulazni signal je dat u pcm formatu, a koeficijenti filtra su izračunati i nalaze se u fajlu `fir.txt`. Koeficijenti FIR filtra se mogu na početku programa kao konstantni niz.

Potrebno je napraviti program koji filtrira ulazni signal u pcm formatu korišćenjem FIR filtra čiji su koeficijenti dati u fajlu `fir.txt`. Inicijalizovati signalni bafer u koji će biti smešteno $L/2$ odbiraka ulaznog signala. U svakom ciklusu obrade se čita po jedan odbirak ulaznog signala i vrši računanje signala na izlazu korišćenjem simetrije ili antisimetrije FIR filtra pomoću formule:

$$y[n] = \sum_{l=0}^{L/2-1} b_l \left(x[n-l] \pm x[n-L+1+l] \right)$$

gde je L broj odbiraka FIR filtra, $x[n]$ ulazni signal, a $h[n]$ odbirak FIR filtra. Nakon obrade jednog podatka se čita novi odbirak iz ulaznog fajla i vrši pomeranje ostalih odbiraka u signalnom baferu kao na slici 24.



Slika 24: Signalni bafer.

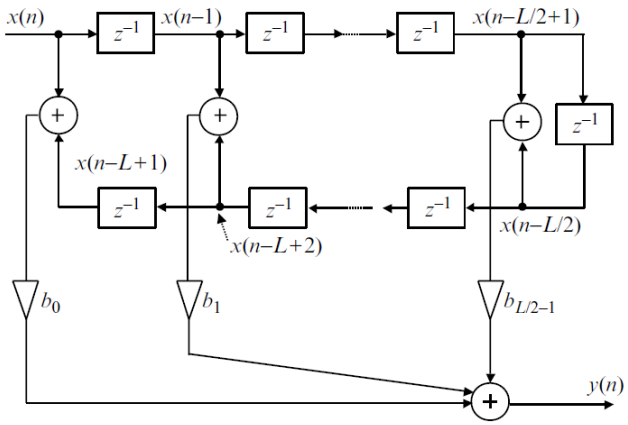
Nakon završetka čitanja ulaznog fajla, potrebno je generisati izlazni fajl u kome se nalaze odbirci izlaznog signala $y[n]$ u pcm formatu. Nazvati izlazni fajl kao ulazni `*_out.pcm`, gde je `*` ime ulaznog fajla (npr. ako je ulazni fajl `ime.pcm`, izlazni je `ime_out.pcm`).

Rezultat izvršavanja programa verifikovati na fajlu `input.pcm` tako što će se pokretanjem skripte u MATLAB-u `compare.m` prikazati rezultati filtriranja u MATLAB-u sa rezultatima filtriranja korišćenjem realizovanog programa. Dati dijagrami bi trebalo da se slažu.

Izmeriti vreme izvršavanja programa.

.....

FIR filtar je simetričan ukoliko su mu oko centralnog odbirka na lokaciji ceil $\left(\frac{L}{2}\right)$ odbirci jednaki, a antisimetričan ako su mu odbirci oko centralnog odbirka jednaki po apsolutnoj vrednosti, ali suprotnog znaka. Na slici 25 je prikazano računanje izlaznih odbiraka kada se koristi osobina simetrije/antisimetrije FIR filtra.



Slika 25: Simetrični FIR filtar.

. . . . . .

Projekat 29 - pravljenje stereo zvuka

Napraviti program koji od dva ulazna PCM fajla koji predstavljaju stereo zvuk jedne iste melodije pravi izlazni fajl u WAV formatu. Odbirci stereo zvuka se zapisuju tako što se upisuju redom odbirci za levi i desni zvučnik. Naime, upisuje se jedan odbirak signala za levi zvučnik, a zatim jedan odbirak za desni zvučnik i tako dok se ne upišu svi odbirci.

Izlazni WAV fajl ima specijalno zaglavlje koje iznosi 44B, nakon čega slede odbirci signala. Zaglavlje WAV falja je kao na linku: `http://www.topherlee.com/software/pcm-tut-wavformat.html`. Obratiti pažnju da između 41-44 bajta u zaglavlju treba ispisati veličinu segmenta koji čine odbirci signala u bajtovima. Odbirci se predstavljaju kao 16-bitni i prepisuju direktno kao u PCM fajlu.

. . . . . .

Projekat 30 - otkrivanje Sublimal poruka

Napraviti program za čitanje skrivenih poruka (*Sublimal messages*, `https://www.youtube.com/watch?v=mFIzn9NrPVo`) u pesmama. Program treba da čita stereo pesmu u WAV formatu i ispisuje je u obrnutom redosledu. Izlazni fajl je takođe u WAV formatu.

Na početku programa potrebno je omogućiti da se izabere ulazni fajl, a zatim segment u pesmi koji je potrebno izvrnuti. Jedna od ponuđenih opcija je potrebno da bude i cela pesma.