Logička zagonetka "Podmornice" u programskom jeziku C

Dumančić, Filip

Undergraduate thesis / Završni rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:176554

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

zir.nsk.hr

Image not found or type unknown

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

ELEKTROTEHNIČKI FAKULTET

Sveučilišni studij / Stručni rad

LOGIČKA ZAGONETKA "PODMORNICE" U PROGRAMSKOM JEZIKU C

Završni rad

Filip Dumančić

Osijek, 2016.

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. PROGRAMSKO OKRUŽENJE C 2

2.1. Uvod u programske jezike 2

2.2. Programski jezik C 4

2.2.1 Povijest programskog jezika C 4

2.2.2 Sintaksa programskog jezika C 5

2.2.3 Primjer programa u C-u 7

3. BATTLESHIP (PODMORNICE) 9

3.1. Uvod u Battleship (podmornice) 9

4. RJEŠAVANJE DIPLOMSKOG ZADATKA 11

4.1. Programski dio rada 11

4.2. Rad programskog dijela 11

4.3. Analiza programskog dijela 15

5. ZAKLJUČAK 18

LITERATURA 19

SAŽETAK 20

ABSTRACT 21

ŽIVOTOPIS 22

1. UVOD

Igra podmornice, izvornog imena Battleships, je jednostavna igra pogađanja za dvoje igrača. Poznata je širom svijeta još od vremena Prvog svjetskog rata. Različite tvrtke su još od 1930-ih godina izdavale svoje verzije ove igre koje su se igrale pomoću papira i olovke. Kasnije, 1967. godine, Milton Bradley izdaje igru u obliku plastične kutije. Prva računalna verzija ove igre pojavila se 1977. godine pod nazivom *Electronic Battleship.*

Ideja i cilj same igre je da igrač pogodi unaprijed postavljenu kombinaciju podmornica drugog igrača. Svaki igrač ima na raspolaganju jednak broj podmornica različite dužine. Prije početka same igre svaki igrač postavlja kombinaciju podmornica za koju misli da će ju protivnik najteže pogoditi. Nadalje, igrač koji pogađa govori koordinate na kojima misli da se nalazi protivnikova podmornica. Ukoliko se na tim koordinatama nalazi podmornica, igrač označava s x pogođenu koordinatu. Igrači igraju naizmjenično svaki potez. Cilj je pogoditi cijelokupnu protivnikovu kombinaciju u što manje pokušaja. Pobjednik je igrač kojem to prvom uspije.

1.1. Zadatak završnog rada

Zadatak završnog rada je realizacija logičke zagonetke "Podmornice" u programskom jeziku C. Ideja je realizirati igru u kojoj će igrač i računalo međusobno pogađati jedno drugom unaprijed postavljene podmornice.

2. PROGRAMSKO OKRUŽENJE C

2.1. Uvod u programske jezike

Mikroprocesor i drugi logički sklopovi koriste programski jezik koji se naziva strojni jezik. Strojni jezik kao instrukcije koristi nizove binarnih riječi koje govore na koji način logički sklopovi trebaju raditi i kako obraditi podatke.

Programer piše program napisan u mnemoničkim oznakama u neku tekstualnu datoteku pomoću editora teksta. Nakon toga, programer poziva program koji prevodi mnemoničke oznake u binarne instrukcije strojnog jezika. Program koji obavlja pretvorbu naziva se asembler (engl. assembler) dok se programski jezik naziva asemblerski jezik ili skraćeno asembler. Program napisan u asemblerskom jeziku naziva se izvorni program (engl. source code). Na taj način pisanje programa postaje proces iz dva koraka. Prvi korak procesa čini pisanje izvornog koda, dok drugi korak procesa čini prevođenje izvornog koda u izvršni program. Tako se programer oslobađa pisanja binarnih instrukcija te se, do neke mjere, dobiva strojna nezavisnost izvornog programa.

Korištenje asemblerskog jezika, odnosno pisanje programa u asemblerskom jeziku, pruža programeru potpunu kontrolu nad svim komoponentama računala, stoga programer mora dobro poznavati arhitekturu samog računala koje piše program te mora kontrolirati sve operacije koje računalo izvršava. Programiranje najjednostavnijih naredbi rezultira složenim i dugačkim programima pa se programiranje u asembleru koristi samo za određene zadaće vezane uz upravljanje hardwareom. Izvorni program napisan u asembleru jedinstven je za arhitekturu svakog tipa računala. Zbog svih tih razloga za većinu programskih zadaća koriste se viši programski jezici.

Zbog gore navedenih manjkavosti i nedostataka razvijeni su viši programski jezici. Neki od viših programskih jezika su: C, *C++*, *Pascal*, *FORTRAN*, *Python*, *Java*. Programski jezici više razine oslobađaju programera potrebe poznavanja arhitekture svakog računala te omogućavaju prenosivost programa između računala različite arhitekture. Samim time omogućuju jednostavnije i brže programiranje. Programi napisani u višim programskim jezicima moraju prije izvođenje proći postupak prevođenja u izvršni kod. To je zadatak prevodioca (engl. compiler). Naredbe višeg programskog jezika prilagođene su tipu podataka s kojim programer

radi te operacijama nad podacima koje želi obavljati. To je osnovna razlika naspram asemblera koji je prilagođen načinu rada logičkih sklopova i mikroprocesora. Zadatak prevodioca je da program koji je napisan u nekom višem programskom jeziku prevede u kod koji se može izvršiti na zadanom računalu. Na taj način se program napisan u nekom višem jeziku može izvršavati na svakom računalu koje ima prevodilac za taj jezik.

Uvod u više programske jezike datira iz razdoblja 60-ih godina 20. stoljeća. Jezici više razine koji su koristili prevoditelje bili su popularno nazvani autokodovi. Primjeri autokodova su *COBOL* i *Fortran*.

2.2. Programski jezik C

2.2.1 Povijest programskog jezika C

Autor ovog programskog jezika je Dennis Ritchie, no značajan doprinos nastanku C-a dali su Ken Thompson, kao autor programskog jezika B i Martin Richards, autor programskog jezika BCPL. Dennis Ritchie stvorio je ovaj programski jezik za rješavanje praktičnih problema kodiranja sistemskih programa i jezgre operacijskog sustava UNIX, koji je praktički u cijelosti napisan u C-u.

Od izdavanja "Programskog jezika C" 1978. godine, svijet računala doživio je veliki napredak. Veliki računalni sustavi postali su još snažniji, a osobna računala dobila su mogućnosti koje se do tada nisu mogle ni naslutiti. Za to se vrijeme mijenjao i sam programski jezik C, iako neznatno, te razvijao sve dalje od svojih začetaka kao jezika UNIX operativnog sistema.

Rastuća popularnost C-a, promjene u jeziku tijekom godina i kreiranje prevoditelja od strane onih kojima nije bitan izgled, kombinirano je s potrebom za preciznijom i suvremenijom definicijom jezika. Godine 1983. American National Standard Institute (ANSI) zasniva udrugu čija je svrha bila napraviti "nedvosmislenu i od računala nezavisnu definiciju C jezika". Rezultat svega je ANSI standard za C.

Kao jedan od najvažnijih jezika u povijesti komercijalne računalne industrije, C je do danas ostao jedini programski jezik prilagođen za sve računalne platforme, od malih sustava pa do mrežnih superračunala. Programi napisani u njemu vrlo su bliski načinu rada hardvera te u načelu zahtijevaju od programera dobro razumijevanje rada procesora, memorije, ulazno-izlaznih sklopova itd. Međutim, rad s registrima procesora i adresiranje memorije apstrahirani su pomoću koncepta varijabli i pokazivača, što uz eksplicitne kontrolne strukture i funkcije znatno olakšava programiranje u odnosu na izravno programiranje u strojnim jezicima.[1]

[1] https://hr.wikipedia.org/w/index.php?title=C+%28programski+jezik%29&oldid=4756921

2.2.2 Sintaksa programskog jezika C

Svi C-programi sastoje se od funkcija i varijabli. Funkcije sadrže instrukcije koje određuju koje će operacije biti izvršene, a varijable služe memoriranju podataka. Izvršavanje programa počinje izvršavanjem funkcije *main* koja mora biti prisutna u svakom programu. Funkcija main svoju zadaću obavlja općenito pozivanjem drugih funkcija. Tako se na primjer poziva funkcija *printf* iz standardne biblioteke, koja je zadužena za ispis podataka na ekranu. Da bismo mogli koristiti funkcije iz standardne biblioteke, kao što su naredbe zadužene za ulaz i izlaz podataka, program započinjemo predprocesorskom naredbom (#include <stdio.h>).[2]

Kada je u pitanju struktura, C je strogo deklarativan programki jezik koji se sastoji od 32 ključne riječi. Osnovu čini funkcija *main*, naredbe pretprocesoru, deklaracija funkcija koje se koriste u programu (ANSI standard) te deklaracija globalnih varijabli i konstanti. Na kraju svake izvršene naredbe potrebno je upisati znak ';'. Početak i kraj svake funkcije označava se vitičastim zagradama '{' (početak funkcije) i '}' (kraj funkcije). Komentari se pišu između oznaka '/*' (početak komentara) i '*/' (kraj komentara).

Programski jezik C u standardu C89 ima 32 ključne riječi (Sl. 2.1.). Ključne riječi nemoguće je koristiti za imenovanje varijabli ili funkcija.

auto	Double	Int	struct
break	else	long	switch
case	enum	static	typedef
char	extern	register	union
const	float	return	unsigned
continue	for	short	void
default	goto	signed	volatile
do	if	sizeof	while

Sl. 2.1. Ključne riječi programskog jezika C

[2] https://hr.wikipedia.org/w/index.php?title=C+%28programski+jezik%29&oldid=4756921

Osnovni skup znakova u programskom jeziku C čine:

- brojevi: 0-9
- velika i mala slova abecede: A-Z, a-z
- posebni znakovi: < = > ? [\ ] ^ _ { | } ~ ! " # % & ' ( ) * + , - . / : ;
- novi red, vertikalni i horizontalni tabulator

Tipovi podataka:

Programski jezik C podržava 5 osnovnih tipova podataka: znakovni (char), cjelobrojni (int), realni (float, double), bez vrijednosti (void). Osim navedenih moguće je definirati i vlastite tipove podataka uporabom naredbe typedef.

Osnovni tipovi podataka:

Tip	Veličina u bitovima	Područje vrijednosti
char	8	-128 do 127
int	16	-32.768 do 32.767
float	32	3.4E-38 do 3.4E+38
double	64	1.7E-308 do 1.7E+308

Tab. 2.1. Osnovni tipovi podataka

Pomoću prilagodnika moguće je promijeniti osnovne tipove podataka tako da bolje odgovaraju potrebama. Kao prilagodnici koriste se sljedeće ključne riječi: signed, unsigned, long, short.

2.2.3 Primjer programa u C-u

Primjer programa za ispis niza

```c
#include <stdio.h>

main()
{
    printf("Hello World\n");
    return 0;
}
```

```
Izlaz programa:
"Hello World"
```

Primjer programa za učitavanje ulaza od korisnika

```c
#include <stdio.h>
main()
{
    int broj;

    printf("Unesi broj: \n");
    scanf("%d", &broj);

    printf("Broj je:  %d\n", broj);

    return 0;
}
```

```
Izlaz programa:
Unesi broj: 11
Broj je: 11
```

Primjer jednostavne *if else* upravljačke naredbe

```c
#include <stdio.h>

main()
{
    int x=5;
    if (x>3)
        printf("x je veći od 3.\n");
    else
        printf("x je manji ili jednak 3.\n");
    return 0;
}
```

```
Izlaz:
x je veći od 3.
```

Primjer petlje: (*for* petlja)

```c
#include <stdio.h>

main()
{
    int x;

    for(x=0; x<5; x++)
    {
        printf("x: %d\n", x);
    }

    return 0;
}
```

```
Izlaz:
X: 0
x: 1
x: 2
x: 3
x: 4
```

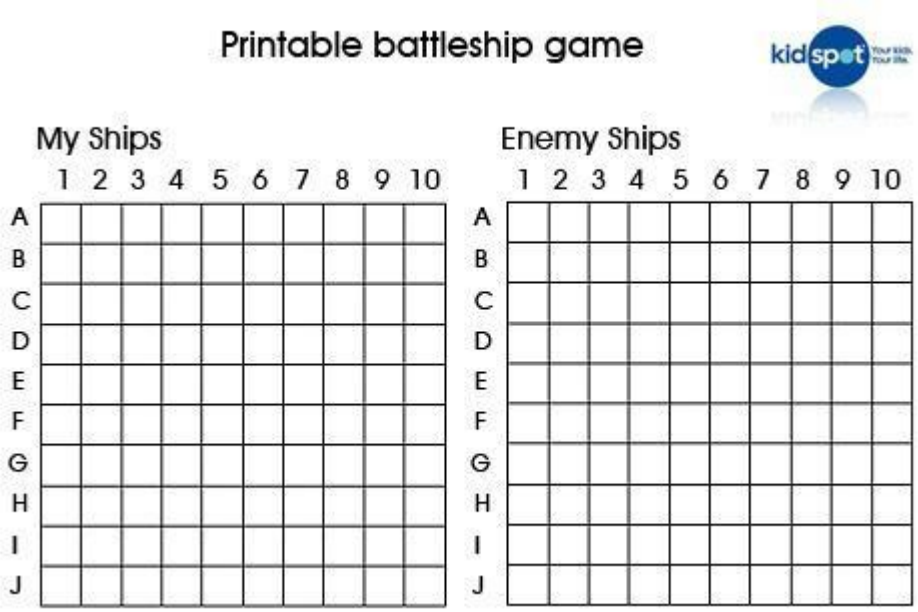
3. BATTLESHIP (PODMORNICE)

Battleship (Podmornice) je igra za dva igrača nastala početkom 19. stoljeća. Različite tvrtke su još od 1930-ih godina izdavale svoje verzije ove igre koje su se igrale pomoću papira i olovke. Kasnije, 1967. godine, Milton Bradley izdaje igru u obliku plastične kutije. Prva računalna verzija ove igre pojavila se 1977. godine pod nazivom *Electronic Battleship.*

3.1. Uvod u Battleship (podmornice)

Prve verzije igre igrale su se pomoću papira i olovke (Sl. 3.1.). Svaki igrač imao je na raspolaganju dva papira s ucrtanim poljem za postavljanje podmornica. Na jedan papir svaki igrač ucrtavao je jednak broj podmornica, ali u vlastitoj kombinaciji koordinata. Nakon što su podmornice ucrtane, igrači su naizmjence pogađali koordinate na kojima smatraju da se nalaze protivnikove podmornice te su na drugo polje označavali pogodak ili promašaj. Pobjednik igre je onaj igrač koji prvi pogodi sve protivnikove podmornice. Zbog velike popularnosti igre, olovku i papir ubrzo su zamijenile modernije verzije u obliku plastičnih kutija s figuricama (Sl. 3.2.). Prvu takvu verziju predstavio je Milton Bradley 1967. godine.

Deset godina kasnije, 1977. godine, Milton Bradley predstavlja računalnu verziju igre zvanu Electronic Battleship. Battleship je jedna od najranijih igara koje su dobile svoju računalnu verziju.



Sl. 3.1. Verzija igre s papirom i olovkom



Sl. 3.2. Plastična kutija s figuricama

4. RJEŠAVANJE DIPLOMSKOG ZADATKA

Ideja igre je na polje 10x10 postaviti kombinaciju podmornica te pogoditi prvi sve protivnikove podmornice. Oba igrača na raspolaganju imaju jednak broj podmornica.

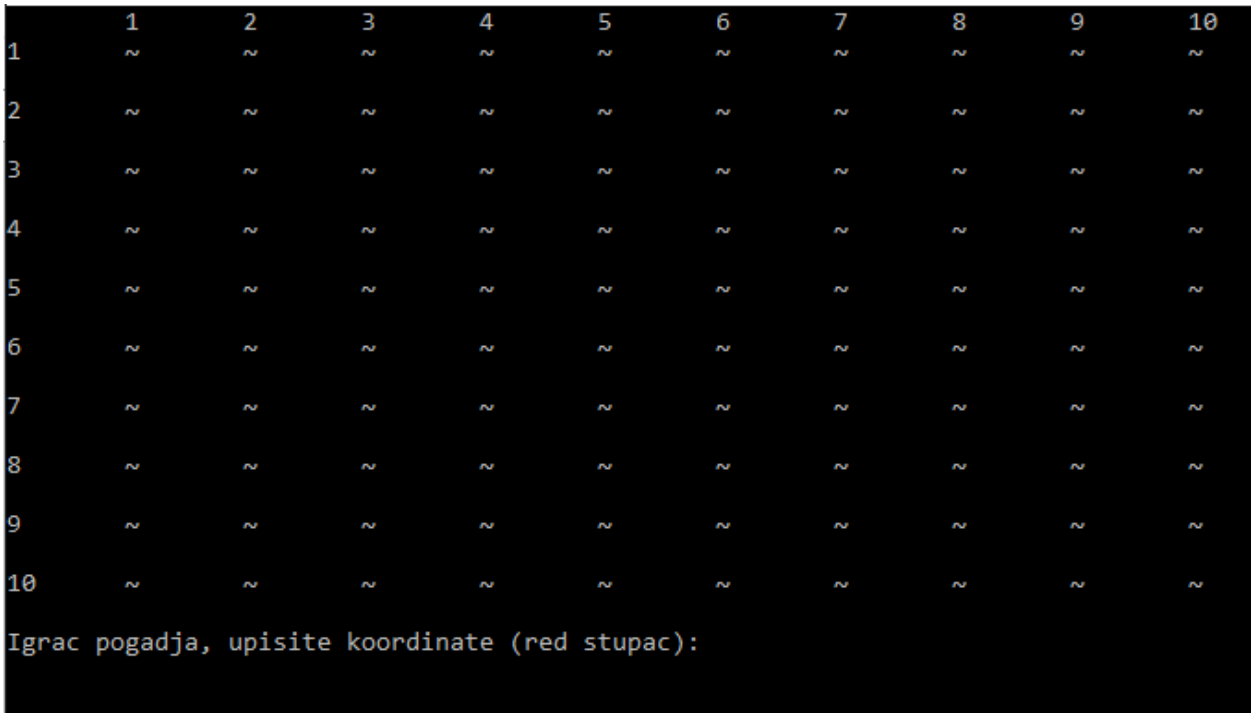
4.1. Programski dio rada

Igra se odvija između igrača i računala. Igrač na papir upisuje svoju kombinaciju podmornica u polje 10x10. Računalo nasumično generira podmornice i također ih postavlja na svoje polje koje je igraču označeno kao 'voda' (~). Na raspolaganju su četiri podmornice 1x4, tri podmornice 1x3, dvije podmornice 1x4 te jedna podmornica 1x6. Igrač prvi upisuje koordinate reda i stupca na kojemu misli da je računalo postavilo svoje podmornice. U slučaju pogotka znak '~' postaje 'X' dok u slučaju promašaja znak postaje '*'. Igrač i računalo igraju naizmjenično. Kada je računalo na potezu, nasumično generira koordinate reda i stupca te ispituje igrača je li došlo do pogotka. U slučaju pogotka računalo na idućem potezu generira nove koordinate, ali ovoga puta preferira koordinate koje su iznad, desno, ispod ili lijevo od prethodno pogođene koordinate. Igrač nastoji pogoditi sve podmornice prije računala i na taj način pobijediti u igri.

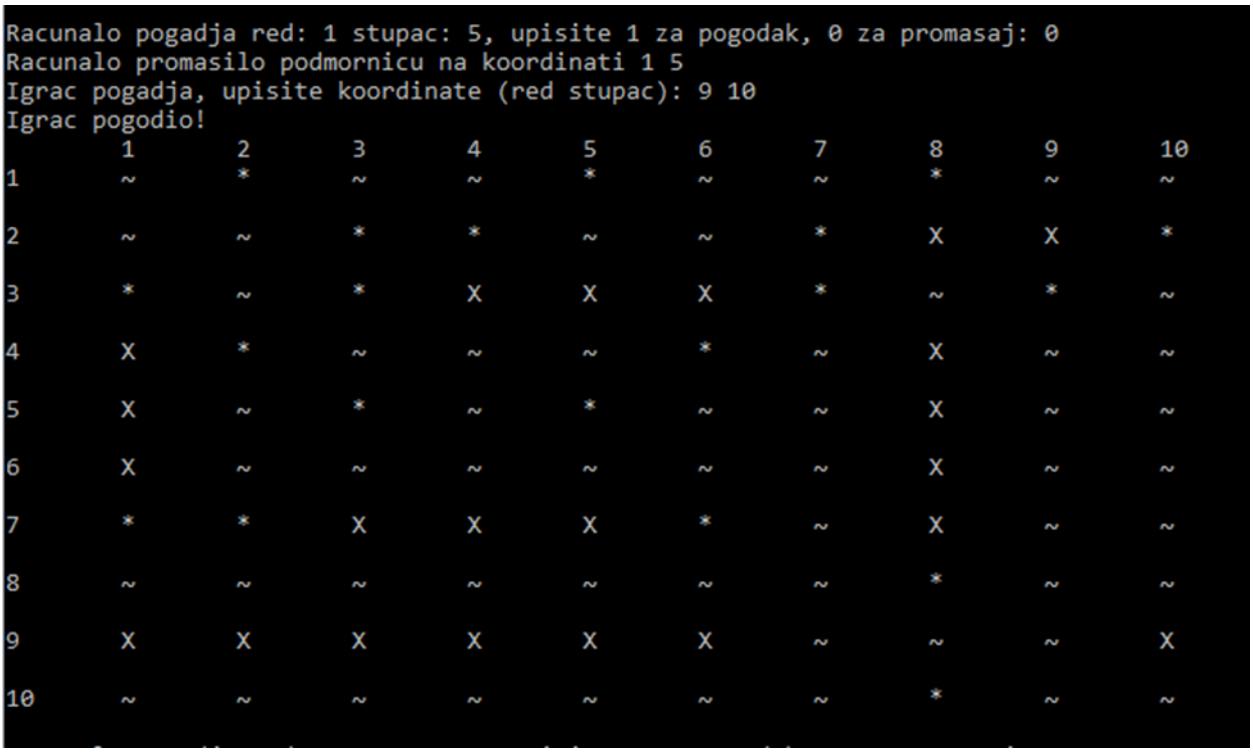
4.2. Rad programskog dijela

Program je napisan u programskom jeziku C te je izrađen kao *Win32 console application*.

Na početku igre igrač igra prvi. Kada je igrač na potezu pojavljuje se polje 10x10 s označenim koordinatama. Na početku su sva polja igraču vidljiva kao voda, označena znakom '~' (Sl. 4.1.). Od igrača se traži da unese koordinate (red i stupac) na kojima misli da se nalazi podmornica.

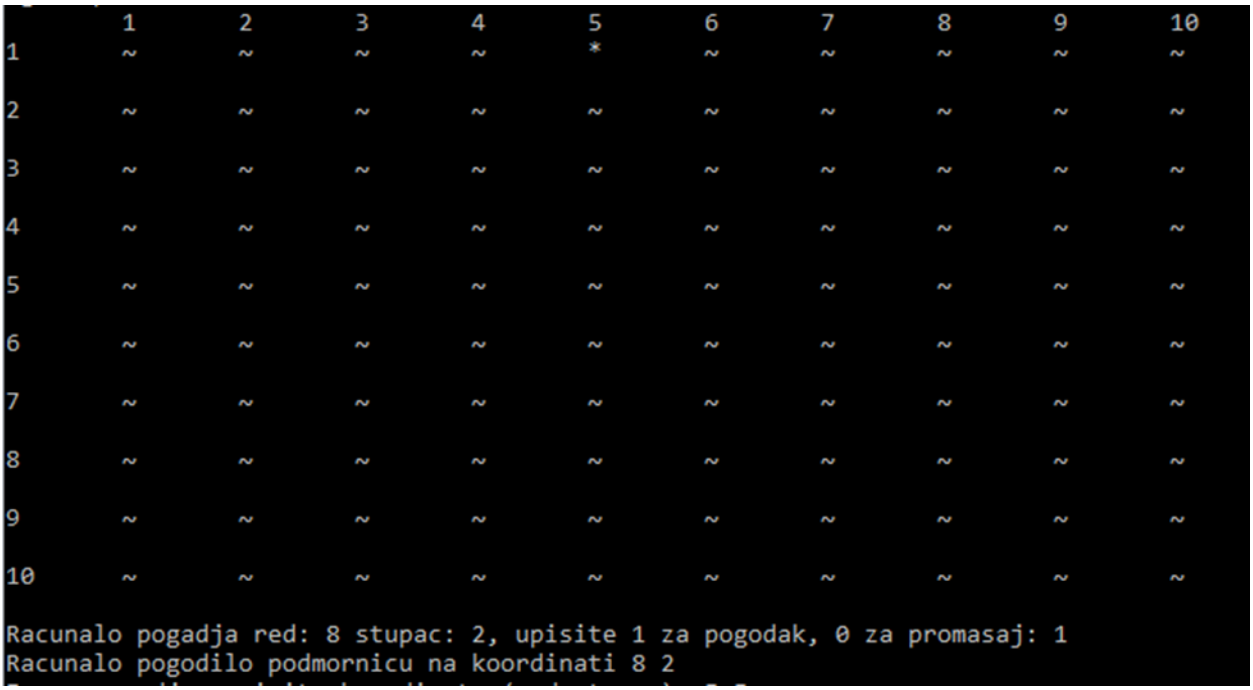


Sl. 4.1. Početak igre



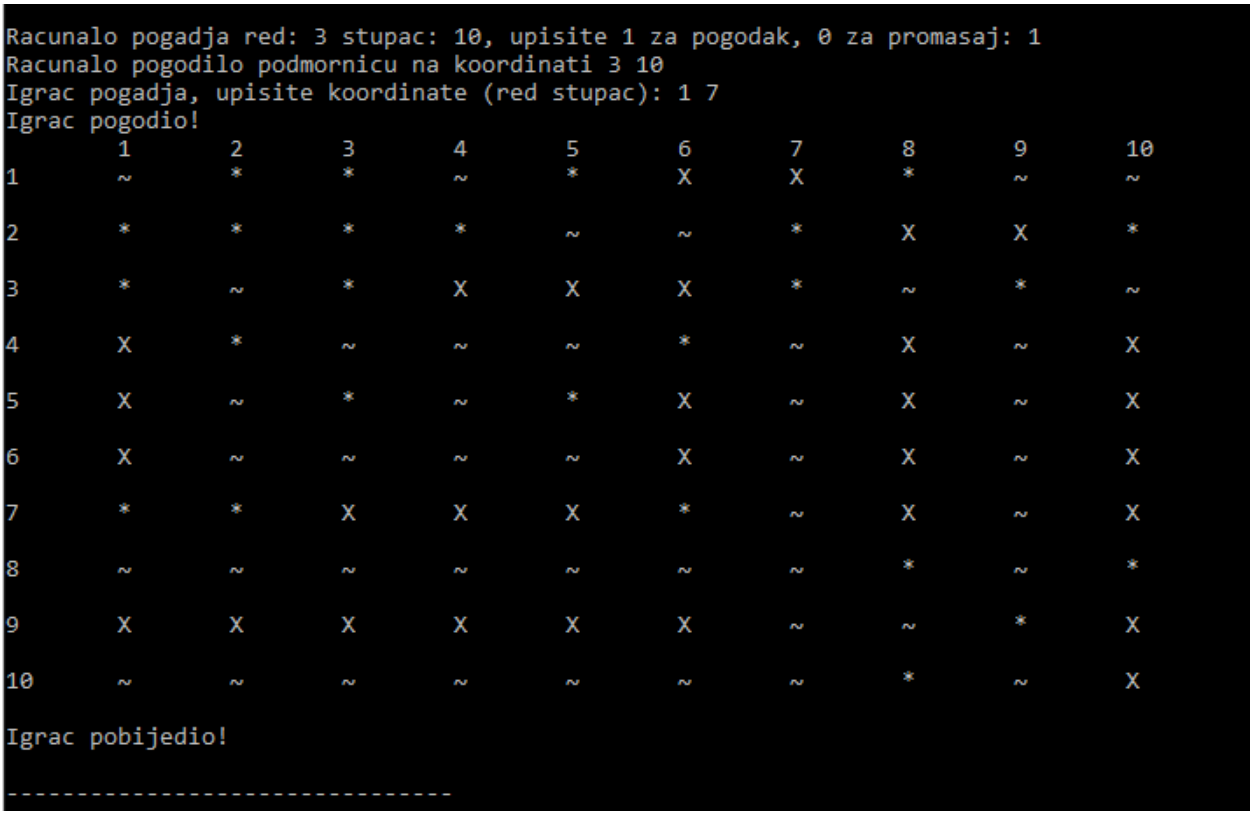
Sl. 4.2. Pogođena/promašena podmornica

Nakon što je igrač završio svoj potez pojavljuje se polje s označenim prethodno odabranim koordinatama. Ukoliko je igrač pogodio podmornicu odabrana koordinata se mijenja u 'X', a ako je igrač promašio podmornicu odabrana koordinata se mijenja u '*'(Sl. 4.2.). Nakon toga na redu je računalo. Računalo nasumično bira dva broja koja predstavljaju red i stupac polja, odnosno koordinate podmornice. Zatim se ispisuje poruka kojom računalo pita igrača nalazi li se podmornica na odabranim koordinatama. Ukoliko je odgovor negativan računalo u svom idućem potezu opet nasumično generira nove koordinate, a ukoliko je odgovor potvrdan računalo će u idućim potezima birati koordinatu koja se nalazi iznad, desno, ispod ili lijevo od prethodno pogođene koordinate. Kada je pogodilo dva uzastopna dijela podmornice, računalo će u istom smjeru pogađati sve dok ne promaši, a kada je promašilo vratit će se na prvu pogođenu koordinatu podmornice i gađati koordinate u suprotnom smjeru od prethodno odabranog. Nakon što opet promaši znači da je pogodilo cijelu podmornicu i tražiti će nove nasumične koordinate (Sl. 4.3.).



Sl. 4.3. Računalo igra

Računalo i igrač naizmjenično biraju koordinate dok jedno ne pogodi sve podmornice. Pobjednik je onaj koji prvi pogodi sve podmornice (Sl. 4.4.).



Sl. 4.4. Kraj igre

4.3. Analiza programskog dijela

Program započinje zaglavljem u kojem su definirane biblioteke (stdio.h, stdlib.h i time.h) koje služe za korištenje ugrađenih funkcija. Nakon toga definirane su varijable koje će se koristiti kasnije u programu. Varijabla 'DEBUG' definirana je tako da služi za testiranje koda. Ukoliko je 'DEBUG' jednak '1' podmornice su vidljive igraču u svrhu testiranja koda. Ukoliko je 'DEBUG' jednak '0' podmornice su skrivene i igra je spremna. Također, definirane su i varijable 'PRAZNO', 'PODMORNICA', 'PROMASAJ' i 'UNISTENA_PODMORNICA'.

Nadalje, slijedi void funkcija 'pokreniPlocu' koja ne vraća nikakvu vrijednost nego služi za stvaranje ploče 10x10 na kojoj će se odvijati igra. Sva polja matrice, odnosno ploče, unutar ove funkcije postavljene su na 'PRAZNO'.

Sljedeća funkcija zove se 'pokaziPlocu'. U njoj je definiran izgled ispisane ploče. Ploča će na marginama imati brojeve od 1 do 10 prema kojima će se tražiti red odnosno stupac. Prilikom ispisa, polje ploče u kojem se nalazi varijabla 'PRAZNO' biti će ispisano kao znak '~' i predstavljat će vodu. Polje ploče u kojem se nalazi varijabla 'PODMORNICA' ispisati će se kao znak '0' i predstavljat će dio podmornice. Međutim, to polje ploče biti će vidljivo igraču samo ako je varijabla 'DEBUG' jednaka '1', u suprotnom će ostati skriveno i bit će ispisano kao znak '~'. Polje ploče na kojem se nalazi vrijednost varijable 'PROMASAJ' biti će ispisano kao znak '*', dok će polje s varijablom 'UNISTENA_PODMORNICA' biti ispisano kao znak 'X'.

Prilikom nasumičnog postavljanja podmornica od strane računala moramo obratiti pažnju u kojem će smjeru biti postavljena podmornica s obzirom na početnu koordinatu (red, stupac). To je rješeno uvođenjem varijable 'smjer'. Ukoliko je vrijednost varijable 'smjer' jednaka '0' smanjivati će se vrijednost varijable 'red' i podmornica će biti postavljena prema gore. Ako je vrijednost varijable 'smjer' jednaka '1' povećavat će se vrijednost varijable 'stupac' i podmornica će biti postavljena prema desno. Kada je vrijednost varijable 'smjer' jednaka '2' povećavat će se varijabla 'red' pa će podmornica biti postavljena prema dolje. I konačno, ukoliko je vrijednost varijable 'smjer' jednaka '3' smanjivat će se vrijednost varijable 'stupac' i podmornica će biti postavljana prema lijevo. Funkcija 'povecajSmjer' služi kako bi se mijenjala određena koordinata u ovisnosti o vrijednosti varijable 'smjer'.

Kako bi podmornica mogla biti postavljena na određeno polje u ploči to polje, i svako polje oko njega, moraju biti slobodni. Za to se brine funkcija 'mjestoSlobodno' koja prima koordinate 'stupac' i 'red' i provjerava jesu li sva potrebna mjesta slobodna.

Funkcija 'moguStavitiPodmornicu' prima varijable 'red', 'stupac', 'smjer' i 'duzina'. Ona provjerava jesu li sva polja dužinom podmornice i oko podmornice slobodna te provjerava je li podmornica unutar ploče. Ukoliko su svi uvjeti zadovoljeni, funkcija javlja kako su sva potrebna polja slobodna te da se na predviđena polja može postaviti podmornica.

Funkcija 'staviPodmornicu' prima varijable 'red', 'stupac', 'smjer' i 'duzina'. Ova funkcija u ovisnosti o zadanoj dužini podmornice određeni broj puta postavlja dio podmornice u određenom smjeru.

Konačno, funkcija 'postaviPodmornice' na ploču postavlja ukupno 31 dio podmornice. Dijelovi podmornica su grupirani tako da čine jednu podmornicu od 6 dijelova (1x6), dvije podmornice od 4 dijela (1x4), tri podmornice od 3 dijela (1x3) te četiri podmornice od 2 dijela (1x2). Svaka podmornica se postavlja na način da se prvo nasumično odredi početno polje i smjer. Ukoliko je zadovoljena funkcija 'moguStavitiPodmornicu', odnosno ukoliko su sva predviđena polja slobodna, postavlja se pripadajuća podmornica. Međutim, ukoliko predviđena polja nisu slobodna, funkcija traži nove nasumične koordinate polja i novi smjer sve dok ne zadovolji zadane uvjete (beskonačna petlja iz koje se ne izlazi dok se ne nađu odgovarajuće koordinate i smjer). Kada je podmornica postavljena prelazi se na iduću sve dok se ne postave sve podmornice. Ovom funkcijom završeno je nasumično postavljanje podmornica od strane računala.

Sljedeće što je trebalo napraviti bila je funkcija u kojoj će računalo pogađati podmornice igrača. Napravljena funkcija zove se 'racunaloIgra'. Nasumično se bira koordinata polja koja prethodno nije pogađana. Ukoliko je odabrana koordinata promašaj ('PROMASAJ') bira se nova nasumična koordinata. Međutim, ukoliko je nasumično odabrana koordinata pogodak, funkcija istražuje tako da u svom sljedećem potezu računalo bira koordinate gore, desno, dolje ili lijevo od prethodno pogođene koordinate. Ako su pogođena dva dijela podmornice jedan do drugog, računalo otkriva smjer u kojem je raspoređena podmornica te pogađa u tom smjeru sve dok ne promaši. Kada je računalo promašilo, vraća se na početnu koordinatu podmornice koju je pogodilo te pogađa polje suprotno od prethodno utvrđenog smjera sve dok opet ne promaši. Kada je promašilo, znači da je pogodilo cijelu podmornicu te traži novu nasumičnu koordinatu. Računalo si sve pogotke i promašaje 'zapisuje' na svoju ploču kako se ne bi dogodilo da je nasumično odabrana koordinata već prethodno bila odabrana.

Na kraju, u glavnoj (*main*) funkciji programa definirane su varijable koje predstavljaju dvije ploče. Prva ploča služi računalu da na nju nasumično rasporedi svoje podmornice koje će igrač

pogađati, dok druga ploča služi računalu za pogađanje igračeve kombinacije podmornica. Obje ploče pokreću se funkcijom 'pokreniPlocu'. Računalo zatim postavlja svoju kombinaciju podmornica preko funkcije 'postaviPodmornice', a onda pokaže ploču igraču (funkcija: 'pokaziPlocu') koji će prvi biti na potezu i pogađati. Nakon toga ulazi se u beskonačnu petlju gdje računalo i igrač naizmjenično pogađaju podmornice. Iz beskonačne petlje izlazi se tek kada računalo ili igrač pogode svaki od 31 postavljenog dijela podmornica. Kada netko od njih pogodi svaki od 31 dijela podmornica znači da je pogodio cijelu protivnikovu kombinaciju i da je pobjedio. Na kraju se ispisuje poruka s pripadajućim pobjednikom.

5. ZAKLJUČAK

U ovom radu realizirana je logička zagonetka "Podmornice" u programskom jeziku C.

Cilj rada bio je objasniti kako logička zagonetka radi te napraviti njezinu programsku realizaciju. U radu je prikazan razvoj programskih jezika te opisan programski jezik u kojem je realiziran rad. Pojašnjena je sintaksa programskog jezika C te su opisani osnovni elementi potrebni za izradu programa u C-u.

Nadalje, opisana je logička zagonetka "Podmornice" te je napravljen kratak osvrt na povijest i razvoj same igre i pravila igranja. Programska realizacija logičke zagonetke napravljena je na način da igrač igra protiv računala.

LITERATURA

[1] Tihomir Čukman, Vlatko Bolt, C/C++ kroz primjere

[2] Dennis M. Ritchie, Brian W. Kernighan, Programski jezik C

[3] M. Jurak, Programski jezik C

[4] https://hr.wikipedia.org/w/index.php?title=C+%28programski+jezik%29&oldid=4756921 (vrijeme pristupa: 19.9.2016.)

[5] https://en.wikipedia.org/wiki/Battleship_(game) (igra "Podmornice") (vrijeme pristupa: 19.9.2016.)

[6] http://boardgames.about.com/od/battleship/a/Rules-of-Battleship.htm (opis i pravila igre) (vrijeme pristupa: 19.9.2016.)

SAŽETAK

Logička zagonetka "podmornice" u programskom jeziku C

Cilj je ovog rada napraviti programsku realizaciju logičke zagonetke "Podmornice" u programskom jeziku C. Napravljen je kratak sažetak o povijesti i načinu rada programskog jezika C te njegovoj sintaksi. Predstavljeni su primjeri jednostavnih funkcija koje se koriste u gotovo svakom programu. Nakon toga, napravljen je kratki osvrt na povijest igre "Podmornice" te su objašnjena pravila igranja. U programskom dijelu rada opisan je način izrade programa popraćen slikama izvedbe. Igra se odvija između igrača i računala. Igrač svoju kombinaciju podmornica piše na papir dok računalo generira nausmičan uzorak podmornica i postavlja ih na polje veličine 10x10. Igrač i računalo naizmjenično pogađaju koordinate podmornica, a pobjednik je onaj koji prvi pogodi sve protivnikove podmornice. Na osnovu stečenog znanja uspješno je napravljena programska izvedba logičke zagonetke.

Ključne riječi: programski jezik C, Battleships, podmornice, kombinacija, program

ABSTRACT

Logic puzzle 'Battleships' in programming language C

The aim of this paper is to develop a program implementation of logic puzzle 'Battleships' in programming language C. Paper starts with brief summary of history and syntax of programming language C. Examples of basic functions used in almost every C program are made. Summary of programming language C is followed by review of history and rules of logic puzzle 'Battleships'. In programming chapter of paper, program development is described along with pictures of program execution. The game takes place between player and computer. The player writes his battleship combination on paper, while computer generates random pattern of battleships and sets them into 10x10 matrix. Player and computer alternately guess each others battleship coordinates. The winner is the one who guesses complete opponents pattern first. Using accuired knowledge, program implementation of logic puzzle is succesfully made.

Keywords: programming language C, battleship, pattern, code, program

ŽIVOTOPIS

Filip Dumančić je rođen je 7. lipnja 1992. godine u gradu Đakovu, Hrvatska. Dolazi iz Semeljaca gdje je završio Osnovnu školu Josipa Kozarca. Paralelno s osnovnom školom upisuje se u školu stranih jezika Memoria u Đakovu gdje završava dva stupnja njemačkog jezika. Nakon završene osnovne škole upisuje Gimnaziju A.G. Matoša u Đakovu gdje završava opći smjer. Po završetku srednje škole upisuje se na Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva s kojeg se ispisuje nakon 2 godine. Godine 2013., kao redovan student, upisuje se na Sveučilište Josipa Jurja Strossmayera u Osijeku Elektrotehnički fakultet. Tijekom školovanja stekao je znanje engleskog jezika. Od 2011. godine posjeduje vozačku dozvolu B kategorije.

potpis: _____