**Blok**

**1**.Od data tri niza odredi koji ima najveći zbir preko rekurzije:

program p1;

type niz=array[1..100] of integer;

var n,i,s,max:integer;

a,b,c:niz;

function zbr(a:niz;n:integer):integer;

*Rekurzivna funkcija za zbir brojeva koja smanjuje* ***n*** *sve do jedinice,a vrednost funkcije je jednaka zbiru poslednjeg člana i zbiru njegovih predhodnika*

var j:integer;

begin

if n=1 then zbr:=a[1]

else zbr:=a[n]+zbr(a,n-1)

end;

begin

readln(n);

for i:=1 to n do read(a[i]);

*Učitavanje nizova*

for i:=1 to n do read(b[i]);

for i:=1 to n do read(c[i]);

***max-****u se dodeljuje zbir prvog niza i redni broj dobija vrednost 1, a zatim se ispituje da li postoji zbir koji je veći, i ako postoji onda on postaje* ***max,*** *a redni broj broj tog niza*

max:=zbr(a,n); s:=1;

if zbr(b,n)>max then

begin max:=zbr(b,n); s:=2; end;

if zbr(c,n)>max then

begin max:=zbr(c,n); s:=3; end;

writeln(s,' ',max); *Ispis*

end.

**2**.Izračunati x/1! - x2/2! + x3/3! - x4/4! +...+ (-1)n+1\*xn/n!

program p1;

var n,k,x,i,p:integer;

s:real;

*rekurzivna funkcija koja računa stepen nekog broja tako što stepen smanjuje za jedan, a vrednost funkcije je jednaka proizvodu broja* ***n*** *i njegovog stepena koji je za jedan manji*

function na(n,k:integer):integer;

begin

if k=0 then na:=1 else na:=n\*na(n,k-1);

end;

*rekurzivna funkcija koja računa faktorijel nekog broja tako što funkcija dobija vrednost proizvoda broja* ***n*** *i fuktorijela broja koji je za 1 manji od njega*

function fac(n:integer):integer;

begin

if n=1 then fac:=1 else fac:=n\*fac(n-1);

end;

begin

*Unos*

*Unos*

readln(x,n);

for i:=1 to n do

begin

***for*** *petlja u kojoj se pomoću funkcija izračunava ono što se traži u zadatku*

p:=na(x,i);

k:=fac(i);

if i mod 2=0 then s:=s-p/k

else s:=s+p/k;

end;

*Ispis*

write(s:0:2);

end.

**3**.Od zadatih **m** brojeva odrediti koji ima najveci zbir cifara preko rekurzije.

program p1;

var m,n,i,max,k,s:integer;

function zbr(n:integer):integer;

*Rekurzivna funkcija za zbir cifara koja smanjuje* ***n*** *sve do nule,a vrednost funkcije je jednaka zbiru poslednje cifre i zbira cifara broja* ***n*** *bez poslednje cifre*

var j:integer;

begin

if n=0 then zbr:=0

else zbr:=n mod 10+zbr(n div 10);

end;

begin

readln(m);

*Učitavanje broja zadatih brojeva i dodela vrednosti* ***max****-u*

max:=0;

for i:=1 to m do

begin

read(n);

*Učitavanje brojeva, pri čemu je* ***k*** *brojač, i ispituje se da li je zbir cifara datog broja veća od* ***max*** *i ako jeste* ***max*** *postaje taj zbir i pamti se* ***k***

k:=k+1;

if zbr(n)>max then

begin

max:=zbr(n);

s:=k;

end;

end;

*Ispis*

writeln(s,'-ti clan ima najveci zbir cifara: ',max);

end.

**4**.Preko DFS-a odrediti da li je zadati čvor povezan sa svim ostalim čvorovima.

program p1;

var i,j,n,q,k:integer;

o:array[1..100] of boolean;

e:array[1..100,1..100] of boolean;

*Procedura* ***DFS*** *dobija pri pozivu vrednost* ***st****,to je čvor od koga treba da počne obilazak i stavi true u* ***o[st]*** *da je* ***st*** *čvor posećen. U* ***for*** *petlji nađe prvi čvor (****i1****) koji je sused od* ***st*** *i uz to je neposećen, pa za taj čvor se poziva* ***DFS[i1] i*** *staviti true u* ***o[i1]*** *da je* ***i1*** *posećen… ovaj postupak nastavlja dok ne dostigne čvor* ***ik*** *koji više nema suseda ili nema suseda koji nije posećen. U ovom trenutku izvršavanja na steku su* ***DFS[st], DFS[i1], DFS[i2],…, DFS[ik]****.Sa steka se skida* ***DFS[ik]*** *jer je ”pala“ i počinje da se izvršava* ***DFS[ik-1]*** *u* ***for*** *petlji za* ***i=ik+1*** *po opisanom postupku*.

procedure dfs(st:integer);

var i:integer;

begin

o[st]:=true;

for i:=1 to n do

if (e[st,i] and not o[i]) then dfs(i);

end;

begin

readln(n,m);

for i:=1 to m do

*Učitavanje gradova koji su povezani i ispunjavanje članova matrice* ***a[i,j]*** *i* ***a[j,i]*** *sa true.*

begin

read(k,j);

a[k,j]:=true; a[j,k]:=true;

end;

begin

*Pokretanje* ***DFS-a***  *za i ako nisu svi gradovi povezani, onda se broji koliko ima ’ostrva’*

dfs(1); c:=1;

for i:=1 to n do

if not o[i] then

begin dfs(i); inc(c); end;

if c=1 then writeln('Svi su povezani')

*Ispis*

else writeln('Nisu svi povezani, ima ',c,' ostrva.');

end.

**5**.Odrediti koliko puta se funkcija za fibonačijev niz poziva rekurzivno.

program p1;

var n,k:integer;

function fib(n:integer):integer;

begin

*funkcija za odredjivanje* ***n-tog*** *člana fibonačijevog niza, pri čemu svaki put kada sa pozove rekurzivno povećamo brojač* ***k***

if (n=1) or (n=2) then fib:=1 else

begin

fib:=fib(n-1)+fib(n-2);

k:=k+2;

end;

end;

begin

read(n);

*Učitavanje broja* ***n,*** *pokretanje funkcije i ispis*

fib(n);

writeln(k,' ',fib(n));

end.

**6**.Napisati rekurzivnu proceduru za Hanojsku kulu.

program p1;

var b:integer;

procedure prebaci(n,sa,na,pom:integer);

begin

if n>0 then

*Rekurzivna procedura radi po principu prebacivanja* ***n-1*** *prstenova na pomoćni,a zatim njihovog vraćanja na početni, a imedju se ispisuju potezi*

begin

prebaci(n-1,sa,pom,na);

writeln(sa,'->',na);

prebaci(n-1,pom,na,sa);

end;

end;

begin

*učitavanje broja i pokretanje procedure za taj broj*

readln(b);

prebaci(b,1,3,2);

end.

**7.** Rekurzija za NZS 2 broja

Program p7;

var a,b,nzd:longint;

procedure f(x,y:longint);

*Procedura koja računa NZS tako što prvo ispituje da li prvi deljiv drugim i ako jeste onda se ispisuje prvi, a ako nije onda se prvi povećava sve dok ne bude*

begin

if x mod y=0 then writeln(x)

else f(x+a,y);

end;

begin

read (a,b);

*Učitavanje brojeva i ispis njihovog NZS-a*

write(f(a,b));

end.

**8**. Napisati rekurzivnu proceduru za izračunavanje nzd 2 broja.

program b5;

var n, m : integer;

function nzd(x, y : integer) : integer;

*rekurzivna funkcija za NZD koja radi tako što računa NZD od ostatka deljenja* ***y*** *sa* ***x*** *i* ***x*** *sve dok* ***x***  *ne bude jednako 0 jer će tada NZD biti jednako* ***y***

begin

if(x = 0) then nzd:=y

else nzd:=nzd(y mod x, x);

end;

begin

readln(n, m);

*Učitavanje brojeva i ispis njihovog NZD-a*

writeln(nzd(n, m));

end.

**9.** Izračunati sumu prvih n članova fibonačijevog niza.

program b6;

var n : integer;

function fib(n : integer) : integer;

*funkcija koja izračunava* ***n-ti*** *član fibonačijevog niza, čiji je princip rada objašnjen u jednom od predhodnih zadataka*

begin

if(n = 1) or (n = 2) then fib:= 1

else fib:= fib(n-1) + fib(n-2);

end;

function zb(n : integer) : integer;

*rekurzivna funkcija koja računa zbir brojeva, koja je takođe objašnjena u jodnom od predhodnih zadataka*

begin

if n > 0 then zb:=fib(n) + zb(n-1)

else zb:=0;

end;

begin

*Učitavanje broja i ispis zbira prvih n članova fibonačijevog niza*

readln(n);

writeln(zb(n));

end.

**10.** Napisati rekurzivne procedure koje računaju binarni ekvivalent broja n u normalnom i obrnutom poretku.

program binarni;

var n:integer;

procedure nor(n:integer);

begin

if n>1 then nor(n div 2);

*Rekurzivne procedure koje ispisuju normalni i inverzni poredak binarnog ekvivalenta broja* ***n*** *tako što u normalnom prvo se poziva funkciju pa ispisuje,a u inverznom obrnuto*

write(n mod 2);

end;

procedure inv(n:integer);

begin

write(n mod 2);

if n>1 then inv(n div 2);

end;

begin

*Učitavanje broja i ispis normalnog i inverznog poretka*

readln(n);

nor(n); writeln; inv(n);

end.

**11.** Napisati rekurziju za n/m na k decimala.

Program p5;

var n,m,k:integer;

*U ovoj proceduri* ***ind*** *služi samo da bi se prvi put ispisalo ‘****,****’ a procedura radi tako što delilac ostaje isti,a deljenik postaje ostatak pri deljenju pomnožen sa 10.* ***k***  *se razumljivo smanjuje za 1*

procedure f(ind:boolean;n,m,k:integer);

begin

if ind then write(n div m,’,’);

else write(n div m);

if (k<>0) then f(false,(n mod m)\*10,m,k-1);

end;

begin

*Učitavanje brojeva i pokretanje procedure koja ispisuje* ***n/m*** *na* ***k*** *decimala*

readln(n,m,k);

f(true,n,m,k);

end.

**12.** Rekurzijom za n nad k ispisati sve binomne koeficijente unetog broja n

Program p6;

var n,i:longint; p:real;

*Funkcija koja računa* ***n*** *nad* ***k*** *rekurzivno po dobro poznatom algoritmu:*

**f:=f(n-1,k-1)\*f(n-1,k)**

*do kog se lako dolazi pomoću Paskalovog trougla*

function f(n,k:longint):real;

begin

if k=0 then f:=1 else

f:=f(n-1,k-1)\*f(n-1,k);

end;

begin

readln(n);

for i:=0 to n do

*Učitava se broj* ***n*** *i ispisuje vrednost funkcije* ***f*** *za svako* ***i*** *u intervalu od* ***0 do n***

begin

p:=f(n,i);

write (p:7:2,' ');

end;

end.