# Задание

Операции многократной точности (т.е. операции с длинными числами).

# Конкретизация задания

Сначала буквам присваивается значение – например: а=23850934, причем все переменные – целые числа, которые по длине не должны превышать 300 знаков. Потом пишется выражение, например f=(a+b)/c+(d+a). При этом деление – это целое от деления делимого на делитель. Операции сложения, вычитания, умножения – обычные арифметические операции, только это операции над длинными числами. Кроме сложения, вычитания, умножения и деления еще могут быть использованы скобки.

# Формат ввода

## Любую строку в поле ввода можно представить в виде

S={C,V}

Где S – константа либо вычисляемое значение. Если это – константа, то С – число в строковом виде, перед которым стоит символ «@», а если S надо найти то S=V, где V – выражение с переменными и числами. В строке не должно быть пробелов и все переменные состоят из одой буквы. Между строками в поле ввода не должно быть пустых строк. Вывод идет аналогично вводу. Пример ввода и вывода можно посмотреть на рисунке

# Интерфейс программы.

Интерфейс программы – ее внешний вид является наглядным и удобным. В верхнем поле ввода пишутся известные величины и формулы, потом жмется кнопка «Расчитать» и в нижнем поле выводятся в алфавитном порядке все переменные. Программа написана в среде программирования Delphi 6. Использовались только стандартные компоненты – TEdit и TButton. Итого на форме размещено поле ввода, поле вывода и кнопка – для запуска процесса расчета. Использование визуальных средств разработки на много ускорило процесс написания программы.


# Структура хранения

Все длинные числа представляются в виде строки, каждый элемент которой – цифра. Если число отрицательное, то первым символом строки будет “-“. Чтобы считать по формуле используем структуру, описанную ниже

type

 dd= record

 lin:array[1..300] of string;

 dl:integer;

 end;

var

 mas:array[1..300] of dd;

В массиве mas хранится уравнение. Каждый элемент массива – отдельное уравнение. В lin хранятся отдельные элементы уравнения. Например mas[3].lin=(‘A’,’=’,’3’,’/’,’(‘,’B’,’-‘,’1’,’)’) соответствует уравнению A=3/(B-1). В первом элементе (mas[i].lin[1]) всегда хранится буква, во втором (mas[i].lin[2]) символ “=”. Такая форма ведения уравнения нужна для удобства вычисления длинных формул. В dl хранится текущее число элементов в уравнении+1. Можно сказать – если dl=4, то в данном элементе хранится не формула, которую надо вычислить, а значение.

Количество строк введенных в поле ввода должно быть не более 300 – это задается размерностью массива mas.

# Используемые процедуры и функции

**zapolnenie**; - заполнение массива mas из поля edt1.

**sum**(a,b:string):string; - сумма чисел, хранящихся в a и b.

**minus**(a,b:string):string; - разность чисел, хранящихся в a и b. minus=a-b

**umn**(a:integer;var xx:string):string; - умножение числа хх на цифру а. Используется в функции umnozen**.**

**umnozen**(a,b:string):string; - произведение чисел, хранящихся в a и b.

**del2**(str:string):string; - делит число str на 2.

**sravnenie**(a,b:string):shortint; - сравнивает число a с числом b. Возвращается результат:

* если а>b, то sravnenie =1
* если а<b, то sravnenie =-1
* если а<b, то sravnenie =0

**delen**(a,b:string):string; - частное от деления a на b. Берется целая часть.

**po\_ch\_num**(ch:string;var znach:string):boolean; - ищет значение буквы в массиве mas. Если значение найдено то возвращается значение «Истина», в противном случае «Ложь». Функция используется для подстановки чисел в выражения. Аргументы: ch – буква, значение которой мы ищем, znach – найденное значение.

**sislo**(ch:char):boolean; - проверяет – является ли символ ch цифрой. Если да – то возвращается «Истина». Функция используется для определения константа или выражения соответствует заданной букве.

**poisk**(z:integer); - процедура для вычисления значения выражения. Она опирается на все предыдущие процедуры и функции. Z – номер вычисляемого значения в массиве mas. Алгоритм ее работы таков: считаем количество открывающихся скобок и если они есть, то начинаем считать с самой вложенной, потом идет умножение и деление, потом сумма и разность. После каждой операции длина уравнения уменьшается и обращение идет опять к процедуре poisk с тем же параметром z.

**vivod**; - вывод найденных значений в поле вывода memo3.

**TForm1.Button1Click**(Sender: TObject); - процедура нажатия на кнопку «Расчет».

**TForm1.Button2Click**(Sender: TObject); - процедура нажатия на кнопку «Задание».

Все функции используют глобальную переменную mas и все процедуры и функции между собой тесно связаны.

Рассмотрим алгоритм работы процедуры **umnozen**

Ввод a,b

Bol=false

да

b=-b

bol=true

b<0

a>=0

нет

нет

да

a=-a

bol=true

a<0

b>=0

нет

да

b=-b

a=-a

a<0

b<0

да

нет

k=length(b)

c=0

tmp=’’

a=0

b=0

r=umn(b[k],a)

r=r+tmp

tmp:=tmp+’0’

c=sum(c,r)

k=k-1

нет

да

k>=1

c=’-‘+c

да

bol=true

# Листинг программы

Выход

umnozen=c

нет

unit Unit1;

interface

uses

 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,

 StdCtrls, Grids;

type

 dd= record

 lin:array[1..300] of string;

 dl:integer;

 end;

 TForm1 = class(TForm)

 Button1: TButton;

 Memo1: TMemo;

 Label1: TLabel;

 Label3: TLabel;

 Memo3: TMemo;

 Button2: TButton;

 procedure Button1Click(Sender: TObject);

 function minus(a,b:string):string;

 function sum(a,b:string):string;

 procedure Button2Click(Sender: TObject);

 private

 { Private declarations }

 public

 { Public declarations }

 end;

var

 Form1: TForm1;

 mas:array[1..300] of dd;

 vs1:integer;{elementov v zn}

 y:integer; {elementov v mas}

implementation

{$R \*.DFM}

**procedure** zapolnenie;

var

 tmp:string;

 j,nn,ll:integer;

 chislo:boolean;

begin

 y:=0;

 for ll:=0 to form1.Memo1.Lines.Count do begin

 tmp:=form1.Memo1.Lines[ll];

 y:=y+1;

 mas[y].dl:=1;

 nn:=0;

 chislo:=false;

 tmp:=ansiuppercase(tmp);

 for j:=1 to length(tmp) do begin

 if tmp[1]='@' then begin

 mas[y].lin[3]:=copy(tmp,4,300);

 mas[y].lin[1]:=tmp[2];

 mas[y].lin[2]:=tmp[3];

 mas[y].dl:=4;

 break;

 end

 else begin

 mas[y].lin[mas[y].dl]:=copy(tmp,nn+1,j-nn);

 mas[y].dl:=mas[y].dl+1;

 nn:=j;

 end;

 end;

 end;

end;

**function** TForm1.sum(a,b:string):string;

var

 tmp,c,tmp2:string;

 i,k,ost,j:integer;

 bol:boolean;

begin

 if ((b[1]='-') and (a[1]<>'-')) then begin

 sum:=form1.minus(a,copy(b,2,300)); exit;

 end;

 if ((a[1]='-') and (b[1]<>'-')) then begin

 sum:=form1.minus(b,copy(a,2,300)); exit;

 end;

 bol:=false;

 if ((b[1]='-') and (a[1]='-')) then begin

 bol:=true; a:=copy(a,2,300); b:=copy(b,2,300)

 end;

 if length(b)>length(a) then begin

 tmp:=b; b:=a; a:=tmp;

 end;

 ost:=0;

 if length(b)<>length(a) then b:='0'+b;

 c:=a;

 j:=length(a);

 tmp2:='';

 for i:=length(b) downto 1 do begin

 tmp2:=tmp2+'0';

 k:=strtoint(c[j])+strtoint(b[i]);

 k:=k+ost;

 ost:=0;

 if k>9 then begin

 ost:=k div 10; k:=k mod 10;

 end;

 c[j]:=inttostr(k)[1];

 j:=j-1;

 end;

 if ost>0 then begin

 tmp2:=inttostr(ost)+tmp2;

 c:=form1.sum(tmp2,c);

 end;

 if bol then c:='-'+c;

 sum:=c;

end;

**function** Tform1.minus(a,b:string):string;

var

 i,la,lb,vv,snos:integer;

 c,tmp:string;

 pom:boolean;

begin

 if ((b[1]='-') and (a[1]<>'-')) then begin

 minus:=form1.sum(a,copy(b,2,300)); exit;

 end;

 if ((a[1]='-') and (b[1]<>'-')) then begin

 minus:=form1.sum(a,'-'+b); exit;

 end;

 if ((b[1]='-') and (a[1]='-')) then begin

 minus:=form1.minus(copy(b,2,300),copy(a,2,300)); exit;

 end;

 c:=a;

 pom:=false;

 la:=length(a); lb:=length(b);

 if lb>la then begin

 pom:=true; c:=b; b:=a; a:=c;

 la:=length(a); lb:=length(b);

 end;

 snos:=0;

 for i:=lb downto 1 do begin

 vv:=strtoint(a[la-lb+i])-strtoint(b[i])-snos;

 snos:=0;

 if vv<0 then begin

 snos:=1; vv:=vv+10;

 end;

 c[la-lb+i]:=inttostr(vv)[1];

 end;

 if snos=1 then begin

 tmp:='';

 for i:=1 to lb do tmp:=tmp+'0';

 tmp:='1'+tmp;

 c:=minus(c,tmp);

 end;

 while ((c[1]='0')and(length(c)>1)) do c:=copy(c,2,300);

 if pom then c:='-'+c;

 minus:=c;

end;

**function** umn(a:integer;var xx:string):string;

var

 i,ost,tmp,dl:integer;

 str:string;

begin

ost:=0;

str:='';

dl:=length(xx);

for i:=dl downto 1 do begin

 tmp:=a\*strtoint(xx[i])+ost;

 if tmp>9 then begin

 str:=inttostr((tmp mod 10))+str;

 ost:=tmp div 10;

 if i=1 then str:=inttostr(ost)+str;

 end

 else begin

 str:=inttostr(tmp)+str;

 ost:=0;

 end;

end;

umn:=str;

end;

**function** umnozen(a,b:string):string;

var

 k,i:integer;

 tmp,c,r:string;

 bol:boolean;

begin

 bol:=false;

 if ((b[1]='-') and (a[1]<>'-')) then begin

 bol:=true; b:=copy(b,2,300);

 end;

 if ((a[1]='-') and (b[1]<>'-')) then begin

 bol:=true; a:=copy(a,2,300);

 end;

 if ((b[1]='-') and (a[1]='-')) then begin

 a:=copy(a,2,300); b:=copy(b,2,300)

 end;

 if ((a='0')or(b='0')) then begin

 umnozen:='0'; exit;

 end;

 k:=length(b);

 c:='0'; tmp:='';

 for i:=k downto 1 do begin

 r:=umn(strtoint(b[i]),a);

 r:=r+tmp;

 tmp:=tmp+'0';

 c:=form1.sum(c,r);

 end;

 if bol then c:='-'+c;

 umnozen:=c;

end;

**function** sravnenie(a,b:string):shortint;

{ если а>b, то сравнение=1

 если а<b, то сравнение=-1

 если а<b, то сравнение=0}

var

 la,lb,i:integer;

begin

 la:=length(a); lb:=length(b);

 if a[1]='-' then begin

 if b[1]='-' then sravnenie:=sravnenie(b,a)

 else sravnenie:=-1;

 exit;

 end;

 if b[1]='-' then begin

 if a[1]='-' then sravnenie:=sravnenie(b,a)

 else sravnenie:=1;

 exit;

 end;

 if lb>la then sravnenie:=-1;

 if lb<la then sravnenie:=1;;

 if la=lb then begin

 for i:=1 to la do begin

 if a[i]>b[i] then begin

 sravnenie:=1;

 exit;

 end;

 if a[i]<b[i] then begin

 sravnenie:=-1;

 exit;

 end;

 end;

 sravnenie:=0;

 end;

end;

**function** del2(str:string):string;

var

 ost,i:integer;

 dr:string;

begin

 ost:=0;

 dr:='';

 for i:=1 to length(str) do begin

 dr:=dr+inttostr((strtoint(str[i])+ost\*10) div 2);

 ost:=((strtoint(str[i])+ost\*10) mod 2);

 end;

 if dr[1]='0' then dr:=copy(dr,2,300);

 del2:=dr;

end;

**function** delen(a,b:string):string;

{delen=round(a/b)}

var

 bol,zzz:boolean;

 pr,tmp,lev,prav,rab:string;

begin

 if b='0' then begin

 showmessage('Íà íîëü äåëèòü íåëüçÿ!');

 delen:='';

 exit;

 end;

 zzz:=false;

 if ((b[1]='-') and (a[1]<>'-')) then begin

 zzz:=true; b:=copy(b,2,300);

 end;

 if ((a[1]='-') and (b[1]<>'-')) then begin

 zzz:=true; a:=copy(a,2,300);

 end;

 if ((b[1]='-') and (a[1]='-')) then begin

 a:=copy(a,2,300); b:=copy(b,2,300)

 end;

 bol:=true;

 lev:='0'; prav:=a; pr:='0'; rab:=a;

 if b<>'1' then

 while (bol) do begin

 tmp:=form1.minus(a,pr);

 if tmp[1]='-' then tmp:=copy(tmp,2,300);

 if (sravnenie(tmp,del2(b))<>1) then break;//bol:=false;

 tmp:=form1.minus(prav,lev); rab:='';

 rab:=form1.sum(lev,del2(tmp));

 pr:=umnozen(b,rab);

 if sravnenie(a,pr)=1 then begin

 lev:=rab;

 end

 else begin

 prav:=rab;

 end;

 end;

 if zzz then rab:='-'+rab;

 delen:=rab;

 if ((rab='-')or(rab=''))then delen:='0';

end;

**function** po\_ch\_num(ch:string;var znach:string):boolean;

var

 bol:boolean;

 i:integer;

begin

 bol:=false;

 for i:=1 to y do begin

 if ((mas[i].dl=4)and(mas[i].lin[1]=ch)) then begin

 bol:=true;

 znach:=mas[i].lin[3];

 end;

 end;

po\_ch\_num:=bol;

end;

**function** sislo(ch:char):boolean;

begin

 if ((ch>='0')and(ch<='9')or(ch='-')) then sislo:=true

 else sislo:=false;

end;

**procedure** poisk(z:integer);

var

 i,k,j,m,k2,zz:integer;

 tmp:string;

 zn:char;

begin

 {snachala ubiraem skobki, potom \*, potom /, potom +, potom -}

k:=0;

 for i:=1 to mas[z].dl-1 do

 if mas[z].lin[i]='(' then k:=k+1;

 if k>0 then begin

 k2:=0;

 for i:=1 to mas[z].dl-1 do begin

 if mas[z].lin[i]='(' then k2:=k2+1;

 if k2=k then begin

 if mas[z].lin[i+2]=')' then begin

 mas[z].lin[i]:=mas[z].lin[i+1];

 for m:=i+1 to mas[z].dl-2 do mas[z].lin[m]:=mas[z].lin[m+2];

 mas[z].dl:=mas[z].dl-2;

 poisk(z);

 exit;

 end

 else begin

 zz:=i+1;

 while mas[z].lin[zz]<>')' do begin

 if ((mas[z].lin[zz]='\*')or(mas[z].lin[zz]='/')) then begin

 if po\_ch\_num(mas[z].lin[zz-1],tmp) then mas[z].lin[zz-1]:=tmp;

 if po\_ch\_num(mas[z].lin[zz+1],tmp) then mas[z].lin[zz+1]:=tmp;

 if (((sislo(mas[z].lin[zz-1][1])))and(sislo(mas[z].lin[zz+1][1]))) then begin

 zn:=mas[z].lin[zz][1];

 if zn='\*' then mas[z].lin[zz-1]:=umnozen(mas[z].lin[zz-1],mas[z].lin[zz+1])

 else mas[z].lin[zz-1]:=delen(mas[z].lin[zz-1],mas[z].lin[zz+1]);

 for m:=zz to mas[z].dl-2 do mas[z].lin[m]:=mas[z].lin[m+2];

 mas[z].dl:=mas[z].dl-2;

 poisk(z);

 exit;

 end;

 end;

 zz:=zz+1;

 end;

 //////// {snachala / i \*, potom + i -}

 zz:=i+1;

 while mas[z].lin[zz]<>')' do begin

 if ((mas[z].lin[zz]='+')or(mas[z].lin[zz]='-')) then begin

 if po\_ch\_num(mas[z].lin[zz-1],tmp) then mas[z].lin[zz-1]:=tmp;

 if po\_ch\_num(mas[z].lin[zz+1],tmp) then mas[z].lin[zz+1]:=tmp;

 if ((sislo(mas[z].lin[zz-1][1]))and(sislo(mas[z].lin[zz+1][1]))) then begin

 zn:=mas[z].lin[zz][1];

 if zn='+' then mas[z].lin[zz-1]:=form1.sum(mas[z].lin[zz-1],mas[z].lin[zz+1])

 else mas[z].lin[zz-1]:=form1.minus(mas[z].lin[zz-1],mas[z].lin[zz+1]);

 for m:=zz to mas[z].dl-2 do mas[z].lin[m]:=mas[z].lin[m+2];

 mas[z].dl:=mas[z].dl-2;

 poisk(z);

 exit;

 end;

 end;

 zz:=zz+1;

 end;

 ////////

 end;

 end;

 end;

 end

 else begin {esli skobok net}

 for zz:=1 to mas[z].dl-1 do begin if ((mas[z].lin[zz]='\*')or(mas[z].lin[zz]='/')) then begin

 if po\_ch\_num(mas[z].lin[zz-1],tmp) then mas[z].lin[zz-1]:=tmp;

 if po\_ch\_num(mas[z].lin[zz+1],tmp) then mas[z].lin[zz+1]:=tmp;

 if ((sislo(mas[z].lin[zz-1][1]))and(sislo(mas[z].lin[zz+1][1]))) then begin

 zn:=mas[z].lin[zz][1];

 if zn='\*' then mas[z].lin[zz-1]:=umnozen(mas[z].lin[zz-1],mas[z].lin[zz+1])

 else mas[z].lin[zz-1]:=delen(mas[z].lin[zz-1],mas[z].lin[zz+1]);

 for m:=zz to mas[z].dl-2 do mas[z].lin[m]:=mas[z].lin[m+2];

 mas[z].dl:=mas[z].dl-2;

 poisk(z);

 exit;

 end;

 end;

 end;

 for zz:=1 to mas[z].dl-1 do begin

 ////////

 if ((mas[z].lin[zz]='+')or(mas[z].lin[zz]='-')) then begin

 if po\_ch\_num(mas[z].lin[zz-1],tmp) then mas[z].lin[zz-1]:=tmp;

 if po\_ch\_num(mas[z].lin[zz+1],tmp) then mas[z].lin[zz+1]:=tmp;

 if ((sislo(mas[z].lin[zz-1][1]))and(sislo(mas[z].lin[zz+1][1]))) then begin

 zn:=mas[z].lin[zz][1];

 if zn='+' then mas[z].lin[zz-1]:=form1.sum(mas[z].lin[zz-1],mas[z].lin[zz+1])

 else mas[z].lin[zz-1]:=form1.minus(mas[z].lin[zz-1],mas[z].lin[zz+1]);

 for m:=zz to mas[z].dl-2 do mas[z].lin[m]:=mas[z].lin[m+2];

 mas[z].dl:=mas[z].dl-2;

 poisk(z);

 exit;

 end;

 end;

///////

 end;

 end;

end;

**procedure** vivod;

var

 i:integer;

begin

 form1.Memo3.Clear;

 for i:=1 to y do

 if mas[i].dl=4 then begin

 form1.Memo3.Lines.Add(mas[i].lin[1]+'='+mas[i].lin[3]);

 end;

end;

**procedure** TForm1.Button1Click(Sender: TObject);

var

 i,j:integer;

begin

zapolnenie;

 for j:=1 to y do

 for i:=1 to y do poisk(i);

 vivod;

end;

**procedure** TForm1.Button2Click(Sender: TObject);

begin

 messagedlg('Îïåðàöèè ìíîãîêðàòíîé òî÷íîñòè',mtinformation,[mbok],0);

end;

end.