Вариант 10

Алгебраическое выражение задано в виде строки. Обеспечить вычисление результата по данному выражению.

Например:

y= «(x^2+5)/(x^3+3\*sin(x)+1)»

при вводе x=0 должен быть результат = 5

**function** weight(s: string): boolean; //проверка является ли символ частью числа

**var**

i: integer;

p: boolean;

nums: **array** [1..11] **of** string := ('1', '2', '3', '4', '5', '6', '7', '8', '9', '0', '.');

**begin**

p := false;

**for** i := 1 **to** 10 **do if** s = nums[i] **then** p := true;

weight := p;

**end**;

**function** chislo(s: string): boolean;

**var**

num: real;

err: integer;

**begin**

chislo := false;

val(s, num, err);

**if** err = 0 **then** chislo := true;

**end**;

**function** math\_operators(s: string): boolean; //проверка является ли строка мат. операцией

**var**

math: **array**[1..5] **of** string := ('+', '-', '\*', '/', '^');

i: integer;

**begin**

**for** i := 1 **to** 5 **do**

**begin**

**if** s = math[i] **then** math\_operators := true;

**end**;

**end**;

**procedure** replace(x: string; **var** s: string); //замена всех x заданным числом

**var**

i, sn, j, lenx: integer;

snew: string;

**begin**

i := 1; sn := 1; lenx := length(x);

**for** i := 1 **to** Length(s) **do**

**begin**

**if** s[i] = 'x' **then**

**for** j := 1 **to** lenx **do**

**begin**

snew += ' ';

snew[sn] := x[j];

sn += 1;

**end**

**else**

**begin**

snew += ' ';

snew[sn] := s[i];

sn += 1;

**end**;

**end**;

s := snew;

**end**;

**procedure** mass(s: string; **var** M: **array of** string); //формирование массива с выражением

**var**

i, j: integer;

num: string;

**begin**

j := 1; num := '';

**for** i := 1 **to** (length(s)) **do**

**begin**

**if** weight(s[i]) **then begin** num += s[i]; **end**

**else begin**

**if** num <> '' **then begin** M[j] := num; num := ''; j += 1; **end**;

**if** ((s[i] = '(') **or** (s[i] = ')')) **then begin** M[j] := s[i]; j += 1; **end**;

**if** math\_operators(s[i]) **then begin** M[j] := s[i]; j += 1; **end**

**else**

**begin**

**if** ((s[i] = 's') **and** (s[i + 1] = 'i') **and** (s[i + 2] = 'n')) **then begin** M[j] := 'sin'; j += 1; **end**;

**if** ((s[i] = 'c') **and** (s[i + 1] = 'o') **and** (s[i + 2] = 's')) **then begin** M[j] := 'cos'; j += 1; **end**;

**if** ((s[i] = 't') **and** (s[i + 1] = 'g')) **then begin** M[j] := 'tg'; j += 1; **end**;

**if** ((s[i] = 't') **and** (s[i + 1] = 'a') **and** (s[i + 2] = 'n')) **then begin** M[j] := 'tan'; j += 1; **end**;

**end**

**end**;

**end**;

**if** ((M[j] = '') **and** (num <> '')) **then** M[j] := num;

**end**;

**procedure** prioritet(M: **array of** string; **var** P: **array of** integer); //формирование массива с приоритетами

**var**

i, pr, j: integer;

**begin**

pr := 0; j := 1;

**for** i := 1 **to** (length(M) - 1) **do**

**begin**

**if** (M[i] = '(') **then begin** pr += 5; P[j] := pr; j += 1; **end**

**else if** ((M[i] = 'sin') **or** (M[i] = 'cos') **or** (M[i] = 'tan') **or** (M[i] = 'tg')) **then begin** pr += 4; P[j] := pr; pr -= 4; j += 1; **end**//////////////////////

**else if** (M[i] = '^') **then begin** pr += 3; P[j] := pr; pr -= 3; j += 1; **end**

**else if** ((M[i] = '\*') **or** (M[i] = '/')) **then begin** pr += 2; P[j] := pr; pr -= 2; j += 1; **end**

**else if** ((M[i] = '+') **or** (M[i] = '-')) **then begin** pr += 1; P[j] := pr; pr -= 1; j += 1; **end**

**else if** (M[i] = ')') **then begin** pr -= 5; P[j] := pr; j += 1; **end**

**else if** (M[i] = '') **then begin** P[j] := 0; j += 1; **end**

**else begin** P[j] := pr; j += 1; **end**;

**end**;

**end**;

**procedure** Mpr(P: **array of** integer; **var** pos: integer); //ищет максимальный приоритет

**var**

i, maxi: integer;

**begin**

maxi := P[1];

**for** i := 1 **to** (length(P) - 1) **do**

**begin**

**if** P[i] >= maxi **then begin** pos := i; maxi := P[i]; **end**;

**end**;

**end**;

**function** ONE(M: **array of** string; pos: integer): boolean; //проверяет нужно ли для вычисления функции 1 переменная

**begin**

**if** ((M[pos - 1] = 'sin') **or** (M[pos - 1] = 'cos') **or** (M[pos - 1] = 'tg') **or** (M[pos - 1] = 'tan')) **then** ONE := true;////////////////////////////

**end**;

**function** TWO(M: **array of** string; pos: integer): boolean; //проверяет нужно ли для вычисления функции 2 переменные

**begin**

**if** ((M[pos] = '+') **or** (M[pos] = '-') **or** (M[pos] = '\*') **or** (M[pos] = '/') **or** (M[pos] = '^')) **then** TWO := true;

**end**;

**procedure** FindOneNum(M: **array of** string; pos: integer; **var** num1: string); //ищет число для вычисления мат. операции с 1 переменной

**var**

i: integer;

**begin**

**if** ONE(M, pos) **then**

**begin**

num1 := M[pos];

**end**;

**end**;

**procedure** FindTwoNum(M: **array of** string; pos: integer; **var** num1, num2: string); //ищет числа для вычисления мат. операции с 2 переменными

**var**

i: integer;

**begin**

**if** TWO(M, pos) **then**

**begin**

num1 := M[pos - 1]; num2 := M[pos + 1];

**end**;

**end**;

**function** CALC(M: **array of** string): boolean; //проверяет есть ли в выражении мат. операции

**var**

i: integer;

**begin**

CALC := false;

**for** i := 1 **to** (length(M) - 1) **do**

**begin**

**if** ((math\_operators(M[i])) **or** (M[i] = 'sin') **or** (M[i] = 'cos') **or** (M[i] = 'tg') **or** (M[i] = 'tan') ) **then**/////////////////////////

CALC := true

**end**;

**end**;

**function** skobka(M: **array of** string): integer; //проверяет, есть ли в выражении скобки

**var**

i, k1, k2: integer;

p: boolean;

**begin**

k1 := 0; k2 := 0; skobka := 0;

**for** i := 1 **to** (Length(M) - 1) **do**

**begin**

**if** M[i] = '(' **then** k1 += 1;

**if** M[i] = ')' **then** k2 += 1;

**end**;

**if** (k1 = k2) **and** (k1 <> 0) **then** skobka := k1

**else if** (k1 = k2) **and** (k1 = 0) **then** skobka := 0

**else** writeln('В введеном выражении неверное количество скобок.');

**end**;

**procedure** DeleteSk(**var** M: **array of** string; **var** P: **array of** integer); //удаляет скобки

**var**

i, j, start: integer;

s: boolean;

**begin**

**for** i := 1 **to** (length(M) - 1) **do**

**begin**

**if** ((M[i] = '(') **or** (M[i] = ')')) **then begin** start := i; s := true; **end**;

**if** (s) **then**

**begin**

s := false;

**for** j := start **to** (length(M) - 2) **do**

**begin**

M[j] := M[j + 1]; P[j] := P[j + 1];

M[j + 1] := ''; P[j + 1] := 0;

**end**;

**end**;

**end**;

**end**;

**function** sinus(num1: string): string; //вычисляет синус

**var**

num1float: real;

err: integer;

**begin**

val(num1, num1float, err);

sinus := FloatToStr(sin(num1float));

**end**;

**function** cosinus(num1: string): string; //вычисляет косинус

**var**

num1float: real;

err: integer;

**begin**

val(num1, num1float, err);

cosinus := FloatToStr(cos(num1float));

**end**;

**function** tangens(num1: string): string; //вычисляет тангенс

**var**

num1float: real;

err: integer;

**begin**

val(num1, num1float, err);

tangens := FloatToStr(tan(num1float));

**end**;

**function** plus(num1, num2: string): string; //сложение

**var**

num1float, num2float: real;

err: integer;

**begin**

val(num1, num1float, err); val(num2, num2float, err);

plus := FloatToStr(num2float + num1float);

**end**;

**function** minus(num1, num2: string): string; //вычитание

**var**

num1float, num2float: real;

err: integer;

**begin**

val(num1, num1float, err); val(num2, num2float, err);

minus := FloatToStr(num1float - num2float);

**end**;

**function** multi(num1, num2: string): string; //умножение

**var**

num1float, num2float: real;

err: integer;

**begin**

val(num1, num1float, err); val(num2, num2float, err);

multi := FloatToStr(num2float \* num1float);

**end**;

**function** step(num1, num2: string): string; //степень

**var**

num1float, num2float: real;

err: integer;

**begin**

val(num1, num1float, err); val(num2, num2float, err);

step := FloatToStr(exp(num2float \* ln(num1float)));

**end**;

**function** devite(num1, num2: string): string; //деление

**var**

num1float, num2float: real;

err: integer;

**begin**

val(num1, num1float, err); val(num2, num2float, err);

devite := FloatToStr(num1float / num2float);

**end**;

**function** solve(pos: integer; M: **array of** string): string; //вычисление функции

**var**

i, j: integer;

rez, num1, num2: string;

**begin**

**if** ONE(M, pos) **then**

**begin**

FindOneNum(M, pos, num1);

**if** M[pos - 1] = 'sin' **then** rez := sinus(num1);

**if** M[pos - 1] = 'cos' **then** rez := cosinus(num1);

**if** M[pos - 1] = 'tg' **then** rez := tangens(num1);

**if** M[pos - 1] = 'tan' **then** rez := tangens(num1);

**end**;

**if** TWO(M, pos) **then**

**begin**

FindTwoNum(M, pos, num1, num2);

**if** M[pos] = '+' **then** rez := plus(num1, num2);

**if** M[pos] = '-' **then** rez := minus(num1, num2);

**if** M[pos] = '\*' **then** rez := multi(num1, num2);

**if** M[pos] = '/' **then** rez := devite(num1, num2);

**if** M[pos] = '^' **then** rez := step(num1, num2);

**end**;

solve := rez;

**end**;

**procedure** ChangeMAS(rez: string; pos: integer; **var** M: **array of** string; **var** P: **array of** integer); //изменение выражения и приоритета

**var**

k: integer;

**begin**

**if** ONE(M, pos) **then**

**begin**

**if** pos > 2 **then**

**begin**

M[pos - 1] := rez; k := 0;

**while** (**not** (chislo(M[pos - 3 - k]))) **do** k += 1;

P[pos - 1] := P[pos - 3 - k];

M[pos] := '#'; P[pos] := 0;

**end**

**else if** pos = 2 **then**

**begin**

M[pos - 1] := rez; P[pos - 1] := 0;

M[pos] := '#'; P[pos] := 0;

**end**;

**end**

**else if** TWO(M, pos) **then**

**begin**

**if** pos > 2 **then begin**

M[pos] := rez; k := 0;

**while** (**not** (chislo(M[pos - 1 - k]))) **do** k += 1;

P[pos] := P[pos - 2 - k];

M[pos + 1] := '#'; P[pos + 1] := 0;

**if** chislo(M[pos - 1]) **then begin** M[pos - 1] := '#'; P[pos - 1] := 0; **end**;

**end**

**else if** pos = 2 **then**

**begin**

M[pos] := rez; P[pos] := P[pos - 1];

M[pos + 1] := '#'; P[pos + 1] := 0;

**if** chislo(M[pos - 1]) **then begin** M[pos - 1] := '#'; P[pos - 1] := 0; **end**;

**end**

**else** //pos=1

**begin**

M[pos] := rez; P[pos] := 0; M[pos + 1] := '#'; P[pos + 1] := 0;

**end**;

**end**

**else** //числа со скобок

P[pos] -= 1;

**end**;

**function** NumSYMB(M: **array of** string): boolean;

**var**

i, k: integer;

**begin**

NumSYMB := false;

**for** i := 1 **to** (length(M) - 1) **do**

**begin**

**if** M[i] = '#' **then** NumSYMB := true;;

**end**;

**end**;

**procedure** deleteSYMB(**var** M: **array of** string; **var** P: **array of** integer);

**var**

i, j, start: integer;

s: boolean;

**begin**

**for** i := 1 **to** (length(M) - 1) **do**

**begin**

**if** (M[i] = '#') **then begin** start := i; s := true; **end**;

**if** (s) **then**

**begin**

s := false;

**for** j := start **to** (length(M) - 2) **do**

**begin**

M[j] := M[j + 1]; P[j] := P[j + 1];

M[j + 1] := ''; P[j + 1] := 0;

**end**;

**end**;

**end**;

**end**;

**procedure** writeMAS(M: **array of** string; P: **array of** integer);

**var**

i: integer;

**begin**

**for** i := 1 **to** (length(M) - 1) **do**

**begin**

write(M[i], ' ');

**end**;

**for** i := 1 **to** (length(P) - 1) **do**

**begin**

write(P[i], ' ');

**end**;

**end**;

**var**

i, j, pos, k: integer;

s, x, snew, rez: string;

massiv: **array of** string;

P: **array of** integer;

**begin**

write('y='); readln(s);

write('x='); readln(x);

replace(x, s);

massiv := **new** string[length(s) + 1];

mass(s, massiv);

P := **new** integer[length(massiv)];

prioritet(massiv, P);

j := skobka(massiv);

**while** (j <> 0) **do**

**begin**

deleteSk(massiv, P); j -= 1;

**end**;

**while** CALC(massiv) **do** //если в выражении есть мат. операции

**begin**

Mpr(P, pos); //находит максимальный приоритет

rez := solve(pos, massiv);

ChangeMAS(rez, pos, massiv, P);

**while** NumSYMB(massiv) **do**

**begin**

deleteSYMB(massiv, P);

**end**;

**end**;

s := massiv[1];

writeln('y=', s);

**end**.

