Министерство образования и науки Российской Федерации   
Федеральное государственное бюджетное образовательное   
учреждение высшего образования   
«Алтайский государственный технический университет им. И.И. Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчёт защищён с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель Астахова Е.В.

«\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2019 г.

Отчёт

по лабораторной работе № 6

«Стековая арифметика»

по дисциплине «Основы программной инженерии»

Студенты группы ПИ 92:

В.М. Шульпов, И.В. Вильк, А. Н. Гулин, Ю.П. Пирязев

Преподаватель доцент, к. п. н. Астахова Е.В.

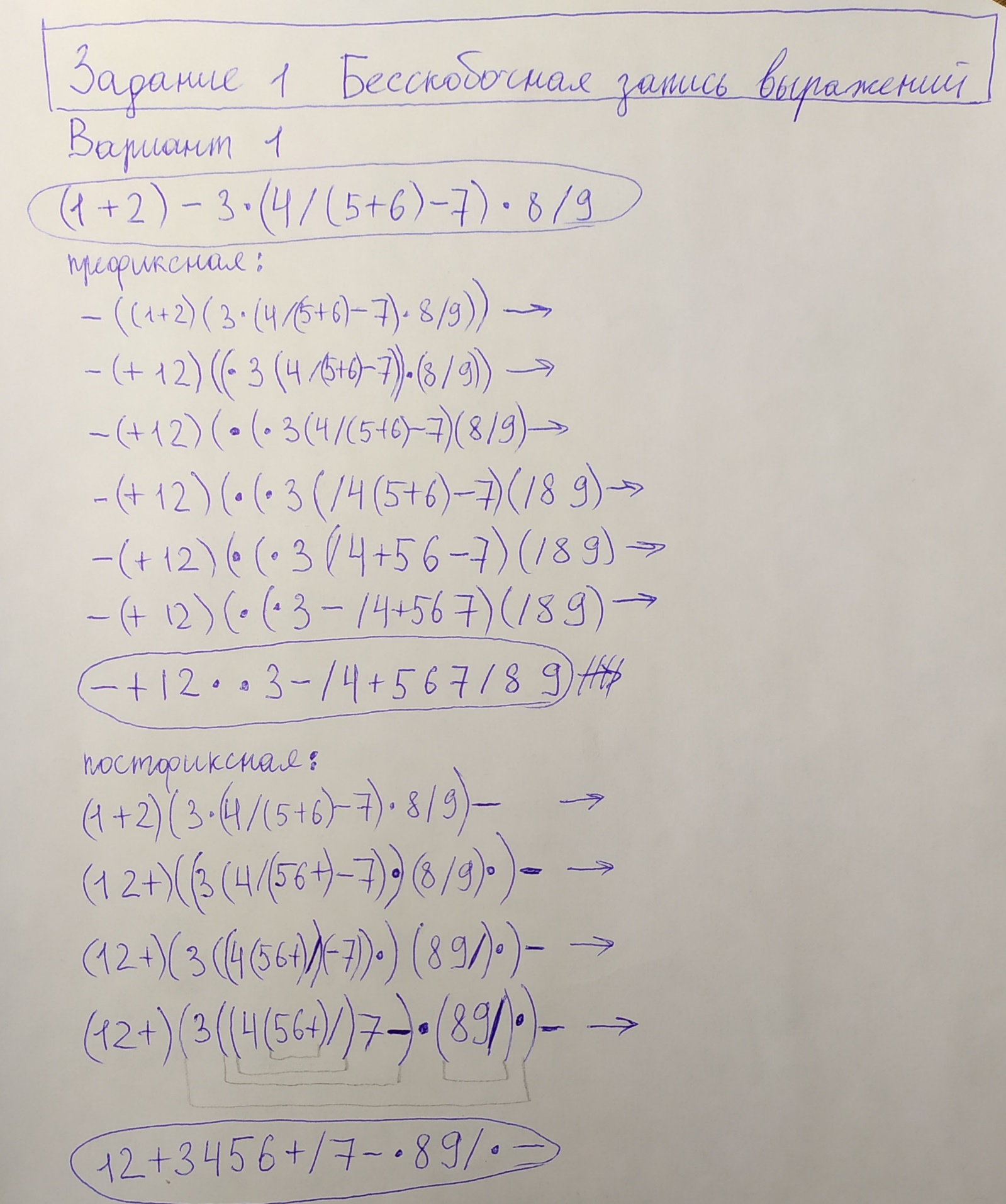
Барнаул 2019

**Задание 1 (вариант 1)**

Инфиксная нотация: (1 + 2) - 3\*(4/(5 + 6) -7)\*8/9

Префиксная нотация: - + 1 2 \* \* 3 - / 4 + 5 6 7 / 8 9

Постфиксная нотация: 1 2 + 3 4 5 6 + / 7 - \* 8 9 / \* -

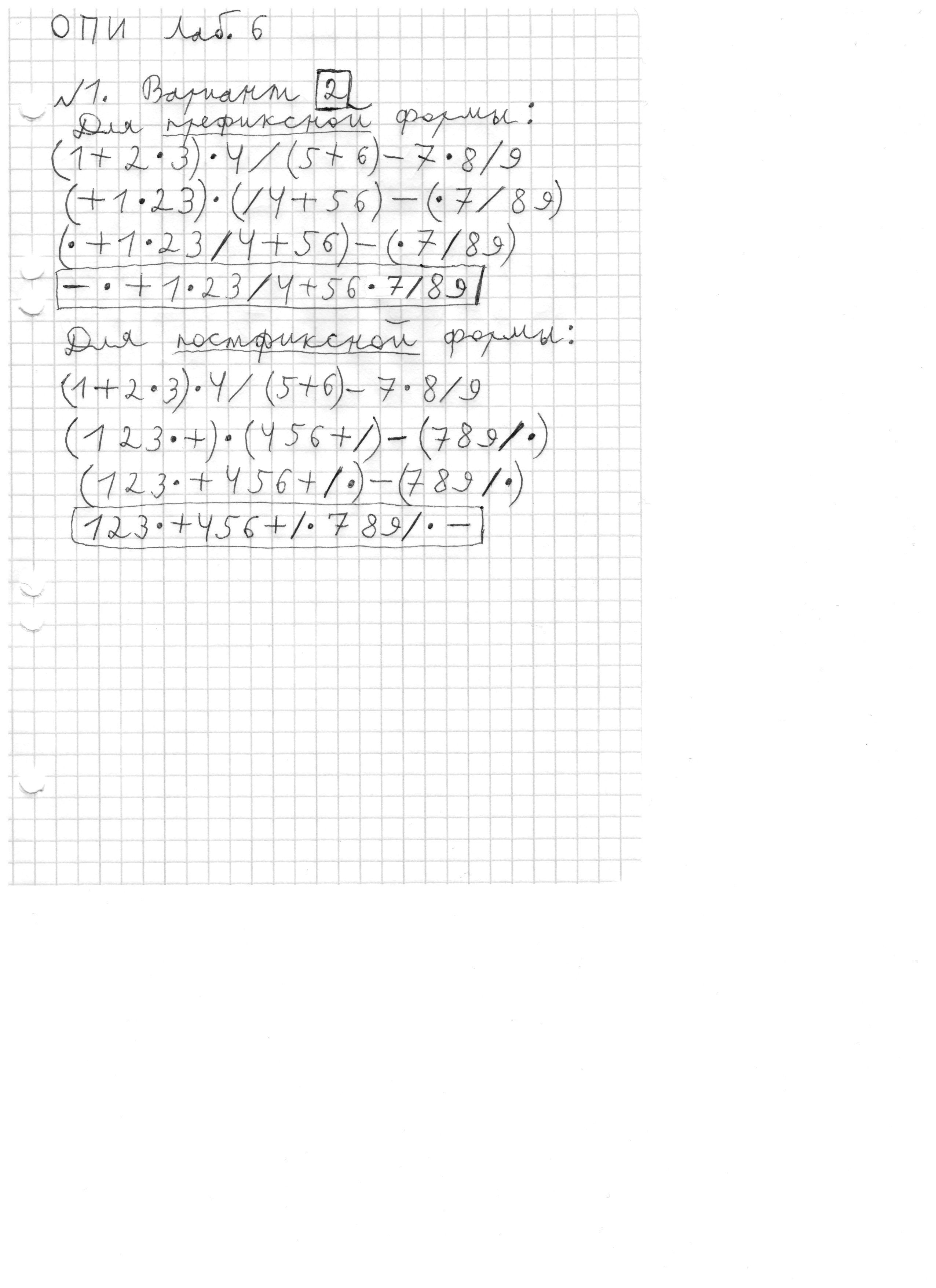
****

**Задание 1 (вариант 2)**

Инфиксная нотация: (1 + 2 \* 3) \* 4 / (5 + 6) - 7 \* 8 / 9

Префиксная нотация: - \* + 1 \* 2 3 / 4 + 5 6 \* 7 / 8 9

Постфиксная нотация: 1 2 3 \* + 4 5 6 + / \* 7 8 9 / \* -

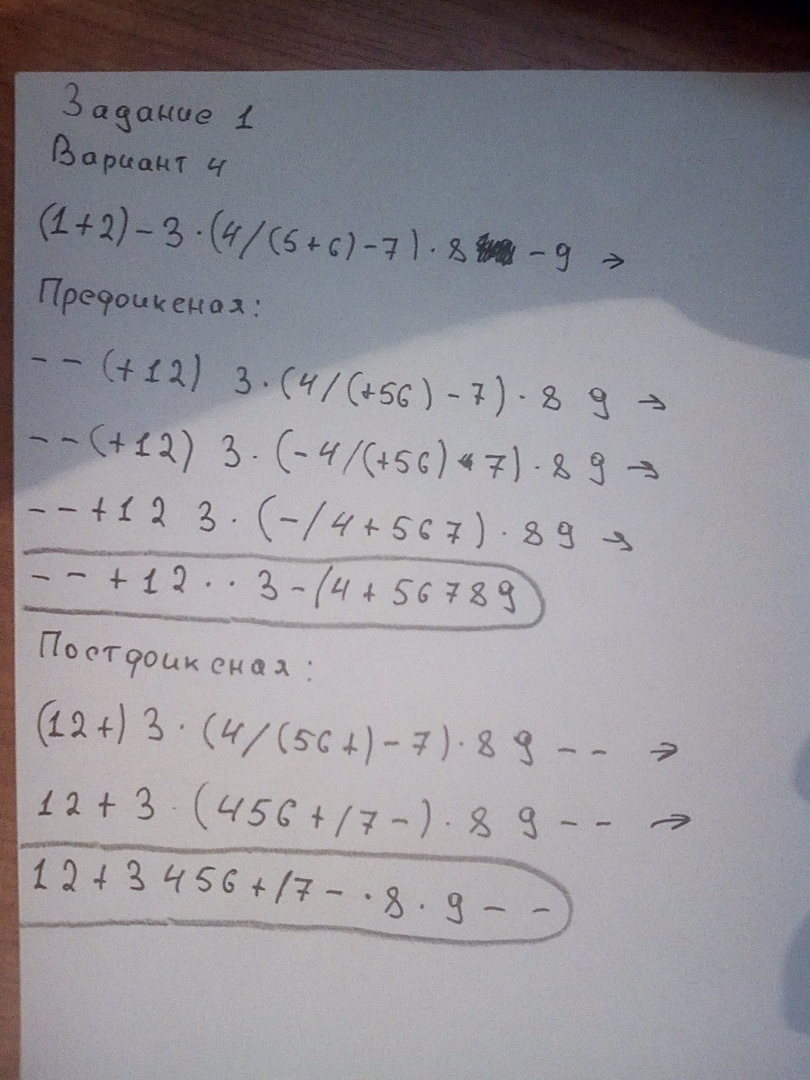
****

**Вариант 4**

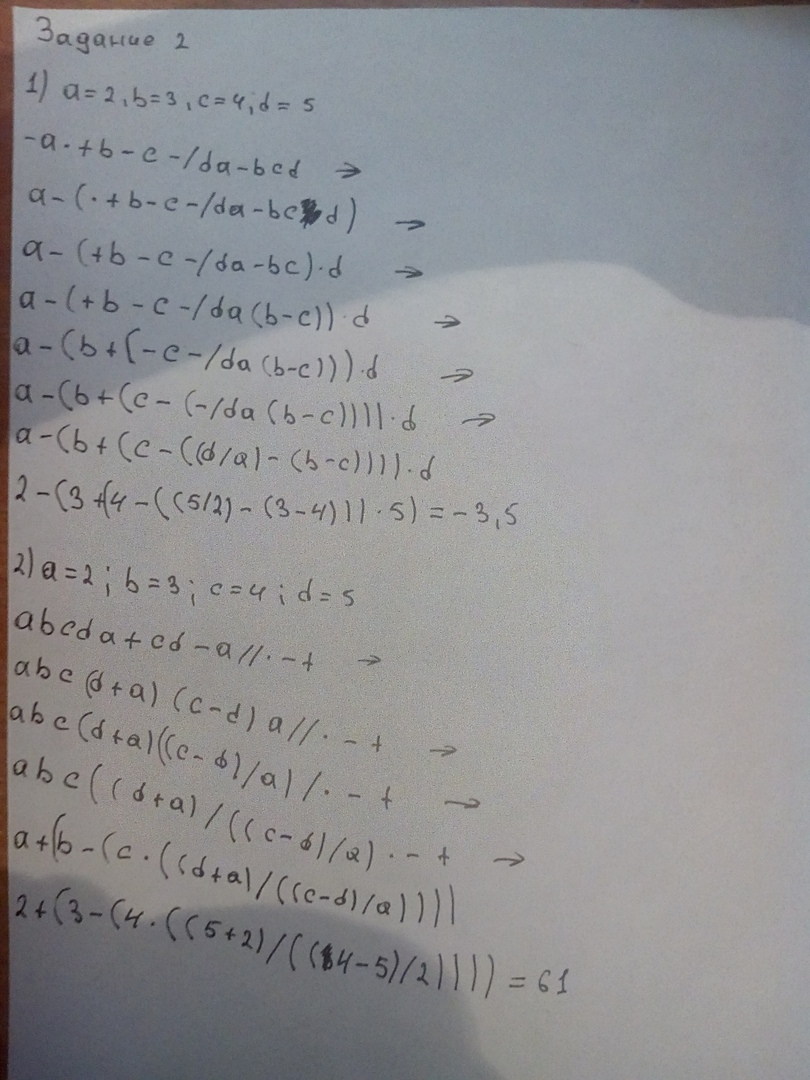
**Инфиксная нотация: (1+2)-3\*(4/(5+6)-7)\*8-9**

**Префиксная нотация: --+12\*\*3-/4+56789**

**Постфиксная нотация: 12+3456+/7-\*8\*9--**



**Задание 2**



**Задача на «4»** Реализовать алгоритм обратной польской записи алгебраического выражения.

// Operator supported: +,-,\*,/,%,^,(,)

// Operands supported: all single character operands

#include"stdafx.h"

#include<conio.h>

#include<ctype.h>

#define MAX 50

typedef struct stack

{

int data[MAX];

int top;

}stack;

int precedence(char);

void init(stack \*);

int empty(stack \*);

int full(stack \*);

int pop(stack \*);

void push(stack \*,int);

int top(stack \*); //value of the top element

void infix\_to\_postfix(char infix[],char postfix[]);

void main()

{

char infix[30],postfix[30];

printf("Enter an infix expression(eg: 5+2\*4): ");

gets(infix);

infix\_to\_postfix(infix,postfix);

printf("\nPostfix expression: %s",postfix);

\_getch();

}

void infix\_to\_postfix(char infix[],char postfix[])

{

stack s;

char x,token;

int i,j; //i-index of infix,j-index of postfix

init(&s);

j=0;

for(i=0;infix[i]!='\0';i++)

{

token=infix[i];

if(isalnum(token))

postfix[j++]=token;

else

if(token=='(')

push(&s,'(');

else

if(token==')')

while((x=pop(&s))!='(')

postfix[j++]=x;

else

{

while(precedence(token)<=precedence(top(&s))&&!empty(&s))

{

x=pop(&s);

postfix[j++]=x;

}

push(&s,token);

}

}

while(!empty(&s))

{

x=pop(&s);

postfix[j++]=x;

}

postfix[j]='\0';

}

int precedence(char x)

{

if(x=='(')

return(0);

if(x=='+'||x=='-')

return(1);

if(x=='\*'||x=='/'||x=='%')

return(2);

return(3);

}

void init(stack \*s)

{

s->top=-1;

}

int empty(stack \*s)

{

if(s->top==-1)

return(1);

return(0);

}

int full(stack \*s)

{

if(s->top==MAX-1)

return(1);

return(0);

}

void push(stack \*s,int x)

{

s->top=s->top+1;

s->data[s->top]=x;

}

int pop(stack \*s)

{

int x;

x=s->data[s->top];

s->top=s->top-1;

return(x);

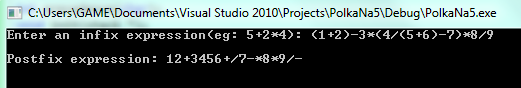
}

int top(stack \*p)

{

return (p->data[p->top]);

}

****

**Задача на «4» (другой вариант реализации)**

#include "stdafx.h"

#include <stdio.h>

#include <conio.h>

#include <locale.h>

#include <windows.h>

#include <string.h>

#define N 1000

#define st 1000 //st - string

char num[st];//глобальная переменная (числа)

char sign[st];//знаки

int sp = -1;//глобальная переменная sp - stack pointer

void addElement(char a) { if (sp != st - 1) num[++sp] = a; }

int \_tmain(int argc, \_TCHAR\* argv[])

{

system("color 70");

setlocale(LC\_ALL, "Rus");

int dcode;//десятичный код символа

char str[N];

char func = 0;//функции - + / \*

int nums = 0;//кол-во чисел в стеке

int signs = 0;//кол-во знаков в стеке

char funcP = 0;//функции - + / \* в скобках (parenthesis-круглые скобки)

int numsP = 0;//кол-во чисел в скобках

int signsP = 0;//кол-во чисел в скобках

for (int i = 0; i < st - 1; i++) num[i] = 0;//инициализация стека нулями

printf("Инфиксная нотация(без пробелов): ");

scanf("%s", str);

int len = strlen(str);

for (int i = 0; i < len; i++)

{

dcode = str[i];

if (dcode == 40) //40-"(", 41 -")"

{

while (dcode != 41)

{

i++;

dcode = str[i];

if (dcode>47 && dcode < 58)//48-57 цифры в десятичной кодировке

{

addElement(str[i]);

numsP++;

if (funcP != 0)

{

addElement(funcP);

numsP--;

funcP = 0;

}

}

if (dcode == 42 || dcode == 43 || dcode == 45 || dcode == 47)//42(\*),43(+),45(-),47(/)

{

if (numsP<= 1)

funcP = dcode;

else addElement(funcP);

}

if (dcode == 41) addElement(func);//добавление ранее записанного знака после скобочного выражения

}

}

if (dcode>47 && dcode < 58)//48-57 цифры в десятичной кодировке

{

addElement(str[i]);

nums++;

if (func != 0)

{

addElement(func);

nums--;

func = 0;

}

}

if (dcode == 42 || dcode == 43 || dcode == 45 || dcode == 47)//42(\*),43(+),45(-),47(/)

{

if (nums<= 1)

func = dcode;

else addElement(func);

}

//printf("sp=%d\n", sp);

for (int i = 0; i <= sp; i++) printf("%c ", num[i]);//вывод стека

printf("\n");

}

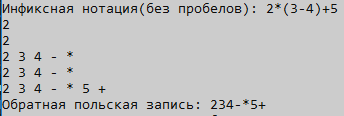
printf("Обратная польская запись: ");

for (int i = 0; i <= sp; i++) printf("%c", num[i]);//вывод стека

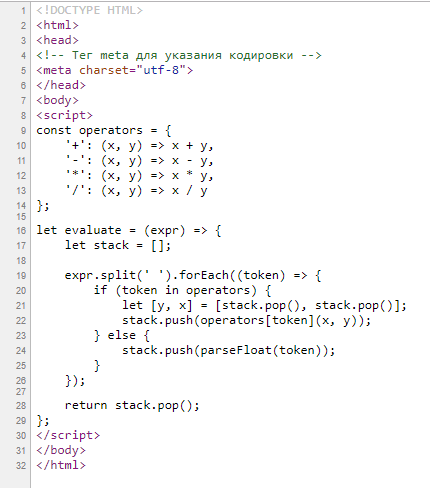
printf("\n");

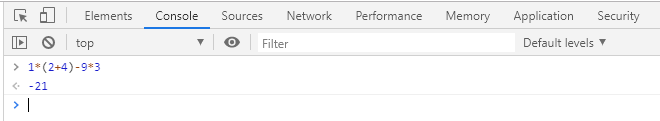
return 0;

}



**Задача на «5» на JavaScript**

****



**Задача на «5» на Python**

import operator

import os

os.system("color F0")

def calc(expr):

OPERATORS = {'+': operator.add, '-': operator.sub, '\*': operator.mul, '/': operator.truediv}

stack = [0]

for token in expr.split(" "):

if token in OPERATORS:

op2, op1 = stack.pop(), stack.pop()

stack.append(OPERATORS[token](op1,op2))

elif token:

stack.append(float(token))

return stack.pop()

print(calc(input()))





