



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

ОТЧЕТ

По лабораторной работе №3

Название : Программирование ветвлений и циклов

Дисциплина: Машинно-зависимые языки и основы компиляции

Студент

ИУ-426

(Группа)

(Подпись, дата)

С.В. Астахов

(И.О. Фамилия)

Преподаватель

(Подпись, дата)

(И.О. Фамилия)

Задание

Вычислить целочисленное выражение:

$$f = \begin{cases} \frac{a+b}{a-b} & \text{если } a * b > 0 \\ \frac{-120}{a * b} & \text{иначе} \end{cases}$$

Схема алгоритма

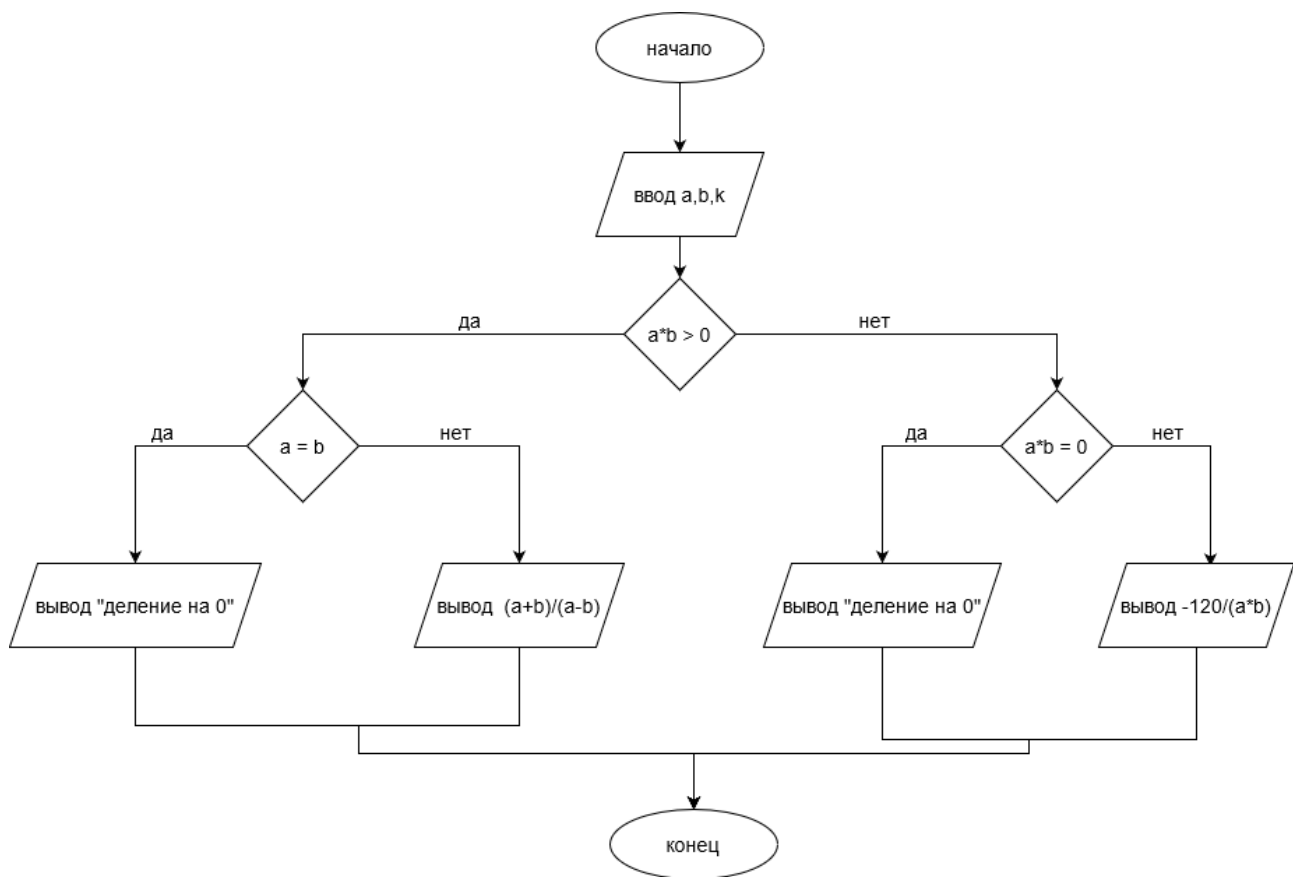


Рисунок 1 — схема алгоритма

Исходный код программы:

```
; Template for console application
.586
.MODEL flat, stdcall
OPTION CASEMAP:NONE
```

```
Include kernel32.inc
```

```
Include masm32.inc
```

```
IncludeLib kernel32.lib
```

```
IncludeLib masm32.lib
```

```
.CONST
```

```
MsgExit DB "Press Enter to Exit",0AH,0DH,0
MsgZeroMulti DB "Division by zero (a*b)",0AH,0DH,0
MsgZeroSub DB "Division by zero (a-b)",0AH,0DH,0
MsgResult1 DB "Result f=(a+b)/(a-b)",0AH,0DH,0
MsgResult2 DB "Result f= -120/(a*b)",0AH,0DH,0
MsgLn DB 0AH,0DH,0
reqA DB 'Input A: ',13,10,0
reqB DB 'Input B: ',13,10,0
```

```
.DATA
```

```
.DATA?
```

```
inbuf DB 100 DUP (?)
buffer DB 10 DUP (?)
outstr DB 10 DUP (?)
A SWORD ?
B SWORD ?
ab SWORD ?
sum_ab SWORD ?
sub_ab SWORD ?
f1_result SWORD ?
f2_result SWORD ?
```

```
.CODE
```

```
Start:
```

```
;
;
; Add you statements
;
```

```
XOR EAX,EAX
```

```
Invoke StdOut,ADDR reqA
Invoke StdIn,ADDR buffer,LengthOf buffer
Invoke StripLF,ADDR buffer
Invoke atoi,ADDR buffer ;result in EAX
mov dword ptr A, EAX
```

```
Invoke StdOut,ADDR reqB
Invoke StdIn,ADDR buffer,LengthOf buffer
Invoke StripLF,ADDR buffer
Invoke atoi,ADDR buffer ;result in EAX
mov dword ptr B, EAX
```

```
mov AX, A
imul B ;DX:AX:=AX*B=A*B
mov ab, AX
```

```

cmp ab, 0
jng negative_ab
    mov AX, A
    cmp AX, B ; a=b?
    jne f1_branch
        Invoke StdOut,ADDR MsgZeroSub
        jmp right_merge
f1_branch:
    mov AX, A
    add AX, B
    mov sum_ab, Ax
    mov AX, A
    sub AX, B
    mov sub_ab, AX

    mov AX, sum_ab
    cwd ;DX:AX = AX
    idiv sub_ab ;AX:=(DX:AX):sub_ab
    mov f1_result, AX

    Invoke StdOut,ADDR MsgResult1
    Invoke dwtoa,f1_result,ADDR outstr
    Invoke StdOut,ADDR outstr
    Invoke StdOut,ADDR MsgLn
right_merge:
    jmp main_merge

negative_ab:
    cmp ab, 0
    jne f2_branch
        Invoke StdOut,ADDR MsgZeroMulti
        jmp left_merge
f2_branch:
    mov AX, -120
    cwd ;DX:AX = AX
    idiv ab ;AX:=(DX:AX):(a*b)
    mov f2_result, AX

    Invoke StdOut,ADDR MsgResult2
    Invoke dwtoa,f2_result,ADDR outstr
    Invoke StdOut,ADDR outstr
    Invoke StdOut,ADDR MsgLn
left_merge:
main_merge:

Invoke StdOut,ADDR MsgExit
Invoke StdIn,ADDR inbuf,LengthOf inbuf

```

```
Invoke ExitProcess,0
End Start
```

Таблица 1 — тесты

Исходные данные	Ожидаемый результат	Полученный результат
5 5	„Division by zero (a-b)“	„Division by zero (a-b)“
7 0	„Division by zero (a*b)“	„Division by zero (a*b)“
0 9	„Division by zero (a*b)“	„Division by zero (a*b)“
8 6	Result f=(a+b)/(a-b)	Result f=(a+b)/(a-b)
	7	7
-6 2	Result f= -120/(a*b)	Result f= -120/(a*b)
	10	10

Контрольные вопросы

1. Какие машинные команды используют при программировании ветвлений и циклов?

Для программирования ветвлений и итерационных циклов используются команды безусловной(jmp) и условной(je, jne, jg, jng, jl, jnl) передачи управления.

Для программирования счетного цикла используется команда loop.

2. Выделите в своей программе фрагмент, реализующий ветвление. Каково назначение каждой машинной команды фрагмента?

```
mov AX, A           ; запись значения A в регистр AX
cmp AX, B           ; сравнение AX(A) и B
jne fl_branch       ; если A != B перейти к вычислениям
...                 ; обработка случая A = B
jmp right_merge     ; перейти к точке слияния ветвей
fl_branch:          ; ветвь вычислений
...
right_merge:        ; точка слияния ветвей
...
```

3. Чем вызвана необходимость использования команд безусловной передачи управления?

Так как без команд передачи управления все команды в программе выполняются последовательно, то без использования команды безусловной передачи управления могут быть выполнены обе ветви алгоритма.

4. Поясните последовательность команд, выполняющих операции ввода-вывода в вашей программе. Чем вызвана сложность преобразований данных при выполнении операций ввода-вывода?

Invoke StdOut,ADDR reqA	; вывод запроса на ввод A
Invoke StdIn,ADDR buffer,LengthOf buffer	; ввод строки
Invoke StripLF,ADDR buffer	; замена символов конца строки нулем
Invoke atol,ADDR buffer	; преобразование строки в число
mov dword ptr A, EAX	; копирование числа из регистра EAX в переменную A

Сложность преобразования данных обусловлена тем, что считанную строку необходимо сначала преобразовать, заменив код конца строки нулем, затем конвертировать в число и затем скопировать в переменную из регистра EAX, учитывая возможную разницу в типе(размере) переменной и размере регистра EAX.

Вывод: в ходе данной лабораторной были изучены команды передачи управления в языке ассемблера, а также реализация с их помощью ветвлений и циклов.