Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: **«**Конструирование программ**»**

**ОТЧЕТ**

к лабораторной работе №5

на тему:

**«СОЗДАНИЕ ПРОСТОЙ ПРОГРАММЫ НА ЯЗЫКЕ АССЕМБЛЕР. Целочисленные арифметические операции. Обработка массивов числовых данных»**

БГУИР 1-40 04 01

|  |
| --- |
| Выполнил студент группы 253505 БЕКАРЕВ Станислав Сергеевич |
|  |
| (дата, подпись студента) |
| Проверил ассистент кафедры информатики РОМАНЮК Максим Валерьевич |
|  |
| (дата, подпись преподавателя) |

Минск 2023

**Цель работы.** Ознакомиться с арифметическими операциями над целочисленными данными, обработкой массивов чисел, ознакомиться с правилами оформления ассемблерных процедур.

В рамках выполнения лабораторной работы должно быть выполнено следующее задание:

Выполнить набор логических побитовых операций над двумя целыми числами, представленными в 10-ной системе счисления.

**Ход работы:** Для выполнения задания был написан программный код на языке Assembler, представленный в листинге №1.

Данный код сохраняет в массив два введенных числа. После чего проводит побитовые операции заданные пользователем. Работа программы с обработкой ошибок ввода показана на рисунке 1.

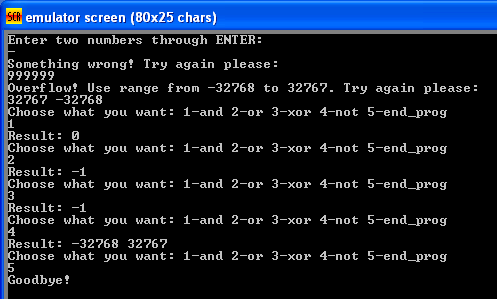


Рисунок 1 – Работа программы.

Листнинг №1 – Исходный код задания

org 100h

start:

call enter

call menu

mov ah, 4Ch

int 21h

enter proc

mov ah, 09h

mov dx, offset enter\_mess

int 21h

mov cx, 2

mov bx, offset array

lp:

mov ah, 0Ah

mov dx, offset two\_num

int 21h

dspace

call check

loop lp

ret

enter endp

check proc

mov minus, 1

mov ax, 0

mov di, offset two\_num + 2

cmp [di], 0Dh

je error

cmp [di], '-'

jne cont1

mov minus, -1

inc di

cmp [di], 0Dh

je error

cont1:

lp1:

cmp [di], 0Dh

jne cont2

mov [bx], ax

inc bx

inc bx

jmp end1

cont2:

sub [di], 30h

cmp [di], 0

jl error

cmp [di], 9

jg error

imul ten

jo overflow

mov dx,[di]

mov dh, 0

push ax

mov ax, dx

imul minus

mov dx, ax

pop ax

add ax, dx

jo overflow

inc di

jmp lp1

error:

inc cx

mov dx, offset wrong\_mess

mov ah, 09h

int 21h

jmp end1

overflow:

inc cx

mov dx, offset overflow\_mess

mov ah, 09h

int 21h

end1:

ret

check endp

menu proc

lp2:

mov ah, 09h

mov dx, offset choose\_mess

int 21h

mov dx, offset choose

mov ah, 0Ah

int 21h

mov di, offset choose + 2

sub [di], 30h

cmp [di], 1

jl error2

cmp [di], 5

jg error2

cmp [di], 1

jne not\_and

call ch\_and

not\_and:

cmp [di], 2

jne not\_or

call ch\_or

not\_or:

cmp [di], 3

jne not\_xor

call ch\_xor

not\_xor:

cmp [di], 4

jne not\_not

call ch\_not

not\_not:

cmp [di], 5

je end2

jmp lp2

error2:

mov dx, offset wrong\_mess

mov ah, 09h

int 21h

jmp lp2

end2:

mov dx, offset goodbye\_mess

mov ah, 09h

int 21h

ret

menu endp

ch\_and proc

mov dx, offset result\_mess

mov ah, 09h

int 21h

mov di, offset array

mov ax, [di]

and ax, [di + 2]

call convert

ret

ch\_and endp

ch\_or proc

mov dx, offset result\_mess

mov ah, 09h

int 21h

mov di, offset array

mov ax, [di]

or ax, [di + 2]

call convert

ret

ch\_or endp

ch\_xor proc

mov dx, offset result\_mess

mov ah, 09h

int 21h

mov di, offset array

mov ax, [di]

xor ax, [di + 2]

call convert

ret

ch\_xor endp

ch\_not proc

mov dx, offset result\_mess

mov ah, 09h

int 21h

mov di, offset array

mov ax, [di]

not ax

call convert

dspace

mov di, offset array

mov ax, [di + 2]

not ax

call convert

ret

ch\_not endp

convert proc

mov minus, 1

mov dx, 0

mov di, offset result\_num

mov cx, 0

call check\_neg

lp3:

div ten

add dx, 30h

push dx

mov dx, 0

inc cx

cmp ax, 0

jne lp3

cmp minus, -1

jne cont3

inc cx

push 002Dh

cont3:

lop3:

pop dx

mov [di], dx

inc di

loop lop3

mov [di], '$'

mov ah, 09h

mov dx, offset result\_num

int 21h

ret

convert endp

check\_neg proc

cmp ax, 32767

jna cont4

neg ax

mov minus, -1

cont4:

ret

check\_neg endp

ten dw 10

minus dw 1

array dw 2 DUP(0)

result\_num db 7 DUP('$')

two\_num db 7,0,7 DUP('$')

choose db 2,0,2 DUP('$')

result\_mess db 0Dh, 0Ah, 'Result: $'

enter\_mess db 'Enter two numbers through ENTER:',0Dh,0Ah,'$'

goodbye\_mess db 0Dh, 0Ah, 'Goodbye!$'

overflow\_mess db 0Dh, 0Ah, 'Overflow! Use range from -32768 to 32767. Try again please:',0Dh,0Ah,'$'

wrong\_mess db 0Dh, 0Ah, 'Something wrong! Try again please:',0Dh,0Ah,'$'

choose\_mess db 0Dh, 0Ah, 'Choose what you want: 1-and 2-or 3-xor 4-not 5-end\_prog',0Dh,0Ah,'$'

space db ' $'

end start

**Выводы:** Была написана программа которая, выполняет набор логических побитовых операций над двумя целыми числами, представленными в 10-ной системе счисления. Были изучены арифметические операциями над целочисленными данными, обработкой массивов чисел, ознакомиться с правилами оформления ассемблерных процедур.