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:

«ЦЕЛОЧИСЛЕННЫЕ АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ. ОБРАБОТКА МАССИВОВ ЧИСЛОВЫХ ДАННЫХ»

БГУИР 1-40-04-01

Выполнил студент группы 253501
Малюш Денис Олегович

(дата, подпись студента)

Проверил старший преподаватель
Ячин Николай Сергеевич

(дата, подпись преподавателя)

Минск 2023

Задание: Ввести массив целых чисел размерностью 30 элементов. Нормализовать значения элементов по максимальному числу в массиве (каждый элемент массива разделить на число с максимальным значением).

Ход работы: на рисунке 1 представлен результат выполнения программы в консоли

Листинг 1 – Исходный код программы задания 1

```
.model tiny
.stack 100h

.data
    inputArrayMessage db "Input number:$"
    normalizedArrayMessage db "Normalized array:$"
    maxElementMessage db "Max Element: $"
    overflowMessage db "ERROR: Overflow$"
    restartMessage db "You want to continue? 1 - Yes 2 - No: $"
    dot db '.$'
    zero db '0$'
    line db 0Dh, 0Ah, '$'
    iterator dw dup('$')
    arr dw 3 dup(0)
    arrOfRemainder dw 3 dup(0)
    newNumber db 5 dup('$')
    number dw 0
    maxElement dw 0

.code
    input macro str
    mov ah, 0Ah
    mov dx, offset str
    int 21h
    endm

    print macro str
    mov ah, 9
    mov dx, offset str
    int 21h
    endm

printNumber macro number
    local convertLoop
    mov ax, number
    mov di, offset buffer + 5
    mov [di], '$'
    dec di

convertLoop:
    mov dx, 0
    mov bx, 10
    div bx
    add dl, '0'
    dec di
```

```
    mov [di], dl
    cmp ax, 0
    jne convertLoop

    mov ah, 9
    mov dx, di
    int 21h
endm

printNumberWithZeroChecking macro number
    local convertLoop
    cmp number, 10
    jg printWithoutAdditionalZero
    print zero
printWithoutAdditionalZero:
    mov ax, number
    mov di, offset buffer + 5
    mov byte ptr [di], '$'
    dec di

convertLoop:
    xor dx, dx
    mov bx, 10
    div bx
    add dl, '0'
    dec di
    mov [di], dl
    cmp ax, 0
    jne convertLoop

    mov ah, 9
    mov dx, di
    int 21h
endm

printArray macro arr
    mov si, offset arr
    mov cl, 0
printingNumber:
    mov ax, word ptr [si]
    mov number, ax
    printNumber (number)
    inc si
    inc si
    inc cl
    cmp cl, 3
    jne printingNumber
endm

printArrayWithRemainder macro arr, arrOfRemainder
    mov si, offset arr
    mov di, offset arrOfRemainder
    mov cl, 0

printingNumber:
    mov ax, word ptr [si]
    mov number, ax
    printNumber number
```

```
    print "."

    mov ax, word ptr [di]
    mov number, ax
    printNumber number

    inc si
    inc si
    inc di
    inc di
    inc cl
    cmp cl, 3
    jne printingNumber
endm

start:
    mov di, 0

addNewNumber:
    print inputArrayMessage
    input newNumber
    print line
    mov ax, 0
    mov bx, 10
    mov si, offset newNumber + 2

numberProcessing:
    mov ch, 0
    mov cl, [si]
    sub cx, 30h

    mul bx
    jo overflow
    add ax, cx
    jo overflow

    inc si
    cmp [si], 0Dh
    jne numberProcessing
    mov arr[di], ax
    inc di
    inc di
    cmp di, 6
    jne addNewNumber
    jmp nextStep

overflow:
    print overflowMessage
    mov ax, 4C00h
    int 21h

nextStep:
    mov si, offset arr
    mov ax, word ptr [si]
    mov cl, 1
    mov maxElement, ax
    inc si
```

```
    inc si

searchMaxElement:
    mov ax, word ptr [si]
    cmp maxElement, ax
    jl lessThanMax
    jmp continue

lessThanMax:
    mov maxElement, ax

continue:
    inc cl
    inc si
    inc si
    cmp cl, 3
    jne searchMaxElement
    print (maxElementMessage)
    printNumber (maxElement)
    print line
    mov si, offset arr
    mov di, offset arrOfRemainder
    mov cl, 0
    mov bx, 100
devide:
    mov ax, word ptr [si]
    mov number, ax
    mov dx, 0
    div maxElement
    mov [si], ax
    cmp al, 1
    je devideContinue

    mov ax, number
    mul bx
    div maxElement
    mov [di], ax

devideContinue:
    inc si
    inc si
    inc di
    inc di

    inc cl
    cmp cl, 3
    jne devide
    print (normalizedArrayMessage)
    print line
    mov si, offset arr
    mov di, offset arrOfRemainder
    mov cl, 0

printingNumber:
    mov ax, word ptr [si]
    mov number, ax
    mov iterator, di
    printNumber number
```

```
    mov di, iterator
    print dot
    mov ax, word ptr [di]
    mov number, ax
    printNumberWithZeroChecking number
    print line
    mov di, iterator

    inc si
    inc si
    inc di
    inc di
    inc cl
    cmp cl, 3
    jne printingNumber

    print restartMessage
    input newNumber
    print line
    mov si, offset newNumber+2
    cmp [si], '1'
    je start

    mov ax, 4C00h
    int 21h
```

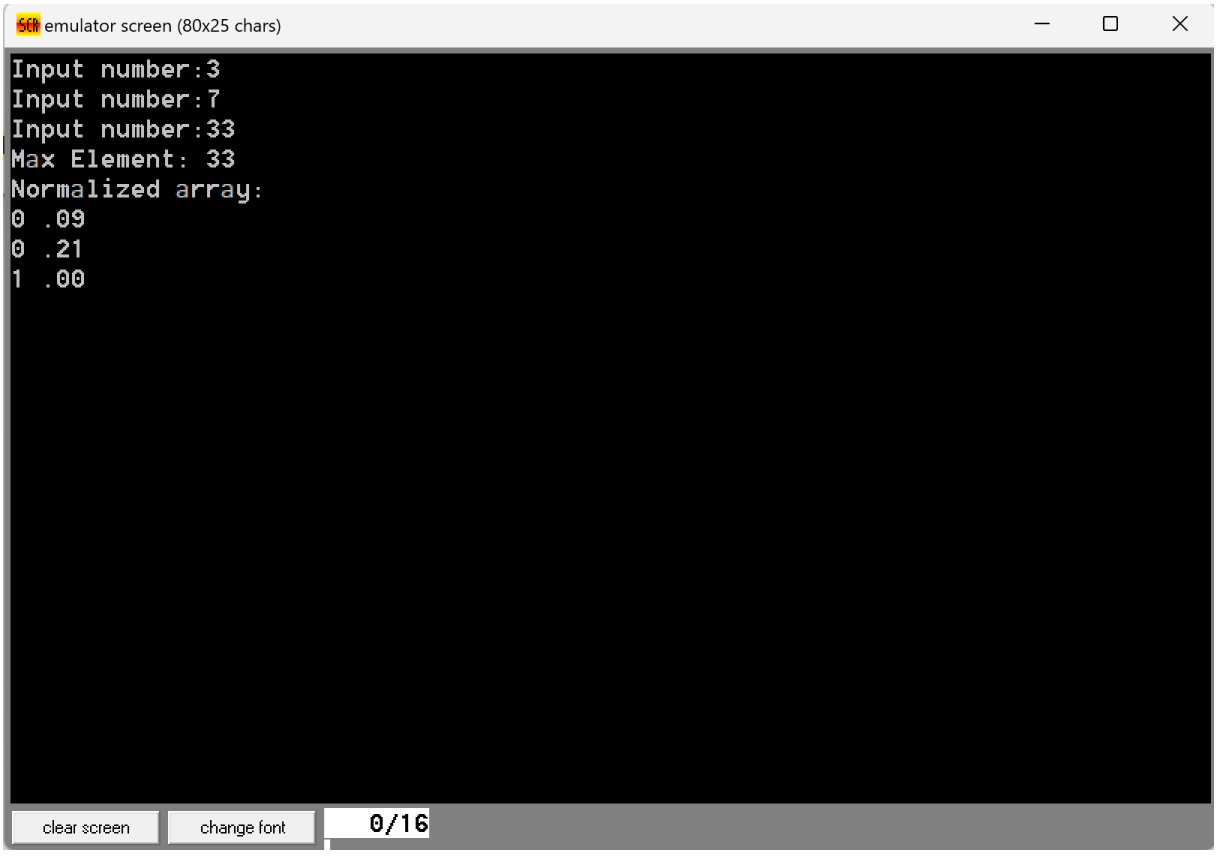

Рисунок 1 – Результат выполнения программного продукта

Выводы: в результате лабораторной работы была выполнена поставленная задача с использованием команд арифметики