МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра № 304 «Вычислительные машины, системы и сети»

Языки ассемблера

Отчет по лабораторной работе

Выполнили студенты группы M3О-311Б-22

Пономарев Н.А.

Проверила доцент Прокимова Т.Г.

Москва 2024 г.

Оглавление

[**Задание** 3](#_Toc182831746)

[**Блок-схема** 4](#_Toc182831747)

[**Ассемблер код** 6](#_Toc182831748)

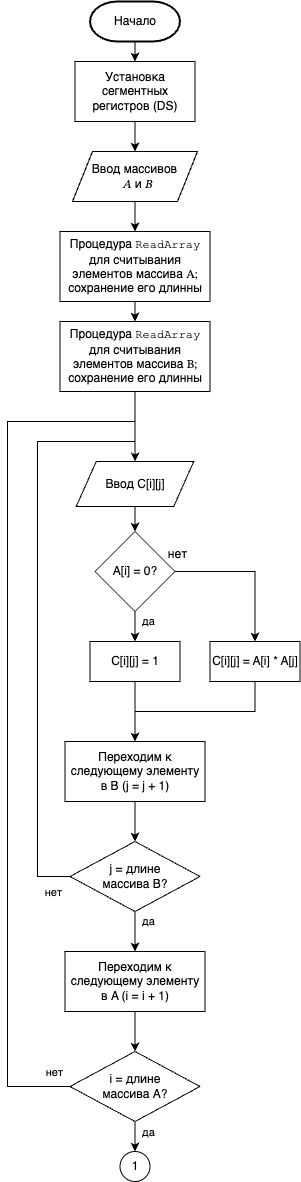
[**Проверочный код на С++** 13](#_Toc182831749)

[**Результаты выполнения программы** 14](#_Toc182831750)

# **Задание:**

Составить программу для ввода двух одномерных массивов А и В. Требуется вычислить значения матрицы С по формуле:

# **Блок-схема:**





# **Ассемблер код:**

; Program dlya rascheta matricy Cmn ispol'zuya massivy A i B

.MODEL SMALL ; Opredelenie modely pamyati - SMALL

.STACK 100H ; Zanyatie steka razmerom 256 bajtov

.DATA

; Soobshcheniya

msgA DB 'Vvedite elementy massiva A (razdelenie probelami): $' ; Soobshchenie dlya vvoda massiva A

msgB DB 'Vvedite elementy massiva B (razdelenie probelami): $' ; Soobshchenie dlya vvoda massiva B

msgC DB 'Rezultat matrica Cmn:', 13, 10, '$' ; Soobshchenie dlya vyvoda rezultata

space DB ' $' ; Probely dlya formatirovaniya vyvoda

newline DB 13, 10, '$' ; Simvoly pereva stroki dlya novogo reda

; Massivy i peremennye

A DW 20 DUP(0) ; Massiv A s maksimal'nym kolichestvom 20 elementov, nachaliye 0

B DW 20 DUP(0) ; Massiv B s maksimal'nym kolichestvom 20 elementov, nachaliye 0

C DW 400 DUP(0) ; Matrica Cmn s maksimal'nym razmerom 20x20 elementov, nachaliye 0

lenA DW 0 ; Fakturnaya dlinna massiva A, nachaliye 0

lenB DW 0 ; Fakturnaya dlinna massiva B, nachaliye 0

; Bufers dlya vvoda strok

inputBuffer DB 100, 0 ; Opredelenie bufara vvoda: maksimal'naya dlinna 100, tekushchaya dlinna 0

DB 100 DUP(0) ; Fakturnoe mesto dlya sokhraneniya vvoda, nachaliye 0

.CODE

START:

MOV AX, @DATA ; Pereemestit' adres segmenta dannykh v registr AX

MOV DS, AX ; Ustanovit' registr DS na adres segmenta dannykh

; Vvod massiva A

LEA DX, msgA ; Zagruska adresu soobshcheniya msgA v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroky na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda soobshcheniya

LEA DI, A ; Ustanovit' registr DI na adres nachala massiva A

CALL ReadArray ; Vyzyvat' podprogrammu ReadArray dlya chteniya massiva A

MOV [lenA], CX ; Sokhranit' kolichestvo prochitannykh elementov massiva A v peremennuyu lenA

; Vyvod novoy stroki posle vvoda massiva A

LEA DX, newline ; Zagruska adresu newline v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroki na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda newline

; Vvod massiva B

LEA DX, msgB ; Zagruska adresu soobshcheniya msgB v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroky na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda soobshcheniya

LEA DI, B ; Ustanovit' registr DI na adres nachala massiva B

CALL ReadArray ; Vyzyvat' podprogrammu ReadArray dlya chteniya massiva B

MOV [lenB], CX ; Sokhranit' kolichestvo prochitannykh elementov massiva B v peremennuyu lenB

; Vyvod novoy stroki posle vvoda massiva B

LEA DX, newline ; Zagruska adresu newline v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroky na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda newline

; Raschet matricy Cmn

MOV SI, 0 ; Ustanovit' registr SI na 0 - nachalo indeksa i dlya massiva A

OuterLoop:

CMP SI, [lenA] ; Sravnit' tekushchiy indeks SI s dlinoy massiva A

JAE CalculationEnd ; Yesli SI >= lenA, pereyti k CalculationEnd

; Podgotovka dlya vnutrennego cikla

MOV CX, 0 ; Ustanovit' registr CX na 0 - nachalo indeksa j dlya massiva B

InnerLoop:

CMP CX, [lenB] ; Sravnit' tekushchiy indeks CX s dlinoy massiva B

JAE NextAi ; Yesli CX >= lenB, pereyti k NextAi dlya sleduyushchego elementa A

; Vychislenie offseta v C: offset = ((SI \* lenB) + CX) \* 2

; Ispol'zuem BX i DI dlya vychisleniya

MOV BX, SI ; Pereemestit' znachenie SI v registr BX

MOV DI, [lenB] ; Pereemestit' znachenie lenB v registr DI

MOV AX, BX ; Pereemestit' BX (SI) v AX

MUL DI ; Vychislenie AX = AX \* DI (SI \* lenB), rezultat v AX

ADD AX, CX ; Dobavit' CX k AX: (SI \* lenB) + CX

SHL AX, 1 ; Umnozit' AX na 2: (offset \* 2) dlya bytovogo offseta

; Vychislenie adresa C[SI][CX]

LEA DI, C ; Pereemestit' adres nachala matrica C v registr DI

ADD DI, AX ; Dobavit' offset k adresu C, poluchit' adres C[SI][CX]

; Zagruska Ai v AX

MOV BX, SI ; Pereemestit' SI v BX

SHL BX, 1 ; Umnozit' BX na 2: BX = SI \* 2 (pozitsiya v massive A)

MOV AX, [A + BX] ; Zagrusit' element A[SI] v registr AX

; Zagruska Bj v DX

MOV BX, CX ; Pereemestit' CX v BX

SHL BX, 1 ; Umnozit' BX na 2: BX = CX \* 2 (pozitsiya v massive B)

MOV DX, [B + BX] ; Zagrusit' element B[CX] v registr DX

; Sravnenie Ai (AX) s 0

CMP AX, 0 ; Sravnit' AX s 0

JNE NonZeroAi ; Yesli AX ne ravno 0, pereyti k NonZeroAi

; Kogda Ai == 0, Cij = 1

MOV WORD PTR [DI], 1 ; Sokhranit' znachenie 1 v C[SI][CX]

JMP StoreComplete ; Pereyti k StoreComplete

NonZeroAi:

; Kogda Ai != 0, Cij = Ai \* Bj

IMUL DX ; Vychislenie AX = AX \* DX (Ai \* Bj), rezultat v AX

; Sokhranit' rezultat v C[SI][CX]

MOV [DI], AX ; Sokhranit' nizshie 16 bit rezultata v C[SI][CX]

StoreComplete:

INC CX ; Uvelichit' CX na 1 - sleduyushchiy indeks j

JMP InnerLoop ; Pereyti k InnerLoop dlya sleduyushchego elementa B

NextAi:

INC SI ; Uvelichit' SI na 1 - sleduyushchiy indeks i

JMP OuterLoop ; Pereyti k OuterLoop dlya sleduyushchego elementa A

CalculationEnd:

; Vyvod rezul'tiruyushchey matricy Cmn

LEA DX, msgC ; Pereemestit' adres soobshcheniya msgC v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroky na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda soobshcheniya

MOV SI, 0 ; Ustanovit' registr SI na 0 - nachalo indeksa i dlya massiva A

PrintOuterLoop:

CMP SI, [lenA] ; Sravnit' SI s dlinoy massiva A

JAE ProgramEnd ; Yesli SI >= lenA, pereyti k ProgramEnd

MOV CX, 0 ; Ustanovit' registr CX na 0 - nachalo indeksa j dlya massiva B

PrintInnerLoop:

CMP CX, [lenB] ; Sravnit' CX s dlinoy massiva B

JAE PrintNextRow ; Yesli CX >= lenB, pereyti k PrintNextRow

; Vychislenie offseta v C: offset = ((SI \* lenB) + CX) \* 2

MOV BX, SI ; Pereemestit' SI v BX

MOV DI, [lenB] ; Pereemestit' lenB v DI

MOV AX, BX ; Pereemestit' BX (SI) v AX

MUL DI ; Vychislenie AX = AX \* DI (SI \* lenB)

ADD AX, CX ; Dobavit' CX k AX: (SI \* lenB) + CX

SHL AX, 1 ; Umnozit' AX na 2: (offset \* 2) dlya bytovogo offseta

; Vychislenie adresa C[SI][CX]

LEA DI, C ; Pereemestit' adres nachala matrica C v registr DI

ADD DI, AX ; Dobavit' offset k adresu C, poluchit' adres C[SI][CX]

; Zagruska Cij v AX

MOV AX, [DI] ; Zagrusit' element C[SI][CX] v registr AX

CALL PrintNumber ; Vyzyvat' podprogrammu PrintNumber dlya vyvoda chisla v AX

LEA DX, space ; Pereemestit' adres probela v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroky na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda probela

INC CX ; Uvelichit' CX na 1 - sleduyushchiy indeks j

JMP PrintInnerLoop ; Pereyti k PrintInnerLoop dlya sleduyushchego elementa B

PrintNextRow:

LEA DX, newline ; Pereemestit' adres novogo reda v registr DX

MOV AH, 09H ; Funktsiya DOS: vyvod stroky na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda novogo reda

INC SI ; Uvelichit' SI na 1 - sleduyushchiy indeks i

JMP PrintOuterLoop ; Pereyti k PrintOuterLoop dlya sleduyushchego elementa A

ProgramEnd:

MOV AH, 4CH ; Funktsiya DOS: vyhod iz programmy

INT 21H ; Vyvolanie DOS prerivaniya dlya zaversheniya programmy

;----------------------------

; Podprogramma dlya chteniya massiva iz vvoda

; Ozhidaet: DI pokazivayet, gde sokhranyat' elementy massiva

; Vozvrashchayet: CX = kolichestvo prochitannykh elementov

ReadArray PROC

; Chtenie strokovogo vvoda

MOV AH, 0AH ; Funktsiya DOS: chtenie buferskogo vvoda

LEA DX, inputBuffer ; Pereemestit' adres inputBuffer v registr DX

INT 21H ; Vyvolanie DOS prerivaniya dlya chteniya vvoda

MOV SI, OFFSET inputBuffer + 2 ; Ustanovit' registr SI na nachalo dannykh vvoda (posle zagolovka)

MOV BX, 0 ; Ustanovit' registr BX na 0 - akkumuliruyuschiy chislo

MOV CX, 0 ; Ustanovit' registr CX na 0 - schetchik elementov

ReadLoop:

MOV AL, [SI] ; Zagrusit' simvol iz bufara vvoda v registr AL

CMP AL, 0DH ; Proverit' esli simvol yavlyaetsya carriage return (vozvrashenie korotki)

JE EndRead ; Yesli da, pereyti k EndRead

CMP AL, ' ' ; Proverit' esli simvol yavlyaetsya probelom

JE StoreNumber ; Yesli da, pereyti k StoreNumber

; Preobrazovanie ASCII v chislo

SUB AL, '0' ; Preobrazovat' ASCII simvol v tsifru (0-9)

CMP AL, 9 ; Proverit' esli tsifra ne prevyshaet 9

JA InvalidInput ; Yesli da, pereyti k InvalidInput

; Akkumulirovanie tsifry v BX

CBW ; Znak-rasshirenie AL v AX (AX = tsifra)

; Umnozhenie akkumuliruyuschego chisla BX na 10

PUSH AX ; Sokhranit' AX (tsifru) na steke

MOV AX, BX ; Pereemestit' BX (akkumuliruyuschiy chislo) v AX

MOV DX, 0 ; Ustanovit' registr DX na 0

MOV BX, 10 ; Ustanovit' registr BX na 10

MUL BX ; Vychislenie AX = AX \* BX (AX \* 10), rezultat v AX

MOV BX, AX ; Pereemestit' rezultat v BX

POP AX ; Vosstanovit' tsifru iz steka v AX

; Dobavlenie tsifry k BX

ADD BX, AX ; Dobavit' tsifru k akkumuliruyuschemu chislu v BX

JMP ContinueRead ; Pereyti k ContinueRead

StoreNumber:

MOV [DI], BX ; Sokhranit' akkumuliruyuschiy chislo v mesto, na kotoroe pokazivaet DI

ADD DI, 2 ; Pereemestit' DI na sleduyushchee mesto (word - 2 byte)

INC CX ; Uvelichit' schetchik elementov na 1

MOV BX, 0 ; Sbrosit' akkumuliruyuschiy chislo v BX na 0

JMP ContinueRead ; Pereyti k ContinueRead

InvalidInput:

; Obrabotka nevalidnogo vvoda (propustit' ili ignorirovat')

JMP ContinueRead ; Pereyti k ContinueRead

ContinueRead:

INC SI ; Uvelichit' SI na 1 - sleduyushchiy simvol v bufere

JMP ReadLoop ; Pereyti k ReadLoop

EndRead:

; Sokhranit' poslednee akkumuliruyuschee chislo nezavisimo ot ego znacheniya

MOV [DI], BX ; Sokhranit' poslednee chislo v mesto, na kotoroe pokazivaet DI

ADD DI, 2 ; Pereemestit' DI na sleduyushchee mesto (word - 2 byte)

INC CX ; Uvelichit' schetchik elementov na 1

ReadArrayEnd:

RET ; Vozvrashchenie iz podprogrammy

ReadArray ENDP

;----------------------------

; Podprogramma dlya vyvoda chisla v AX

PrintNumber PROC

PUSH AX ; Sokhranit' AX na steke

PUSH BX ; Sokhranit' BX na steke

PUSH CX ; Sokhranit' CX na steke

PUSH DX ; Sokhranit' DX na steke

MOV CX, 0 ; Ustanovit' registr CX na 0 - schetchik tsifr

MOV BX, 10 ; Ustanovit' registr BX na 10 - osnovnaya dlya deleniya

CMP AX, 0 ; Proverit' esli AX >= 0

JGE PN\_NotNegative ; Yesli da, pereyti k PN\_NotNegative

; Esli AX otritsatel'nyy, obrabotka znaka

MOV DL, '-' ; Pereemestit' simvol '-' v registr DL

MOV AH, 02H ; Funktsiya DOS: vyvod simvola v DL na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda znaka '-'

NEG AX ; Preobratit' AX v otritsatel'noe znachenie

PN\_NotNegative:

; Preobrazovanie chisla v tsifry

PN\_ConvertLoop:

XOR DX, DX ; Obnulit' DX

DIV BX ; Delit' AX na BX (10): AX = AX / 10, DX = ostatok

PUSH DX ; Sokhranit' ostatok (tsifru) na steke

INC CX ; Uvelichit' CX na 1 - schetchik tsifr

CMP AX, 0 ; Proverit' esli AX != 0

JNE PN\_ConvertLoop ; Yesli da, pereyti k PN\_ConvertLoop

; Vyvod tsifr

PN\_PrintLoop:

POP DX ; Vosstanovit' tsifru iz steka v DX

ADD DL, '0' ; Preobrazovat' tsifru v ASCII simvol

MOV AH, 02H ; Funktsiya DOS: vyvod simvola v DL na ekran

INT 21H ; Vyvolanie DOS prerivaniya dlya vyvoda tsifry

LOOP PN\_PrintLoop ; Dekrementit' CX i proverit' esli CX != 0, yesli da - pereyti k PN\_PrintLoop

POP DX ; Vosstanovit' DX iz steka

POP CX ; Vosstanovit' CX iz steka

POP BX ; Vosstanovit' BX iz steka

POP AX ; Vosstanovit' AX iz steka

RET ; Vozvrashchenie iz podprogrammy

PrintNumber ENDP

END START ; Konechnoe mesto programmы, nachalo - START

# **Проверочный код на С++:**

#include <iostream>

#include <vector>

#include <sstream>

int main() {

std::vector<int> A, B;

std::string line;

int num;

std::cout << "Enter elements of array A (separated by spaces): ";

std::getline(std::cin, line);

std::istringstream streamA(line);

while (streamA >> num) {

A.push\_back(num);

if (A.size() >= 20) break;

}

std::cout << "Enter elements of array B (separated by spaces): ";

std::getline(std::cin, line);

std::istringstream streamB(line);

while (streamB >> num) {

B.push\_back(num);

if (B.size() >= 20) break;

}

std::vector<std::vector<int>> C(A.size(), std::vector<int>(B.size()));

for (size\_t i = 0; i < A.size(); ++i) {

for (size\_t j = 0; j < B.size(); ++j) {

if (A[i] != 0) {

C[i][j] = A[i] \* B[j];

}

else {

C[i][j] = 1;

}

}

}

std::cout << "Resulting matrix Cmn:\n";

for (size\_t i = 0; i < C.size(); ++i) {

for (size\_t j = 0; j < C[i].size(); ++j) {

std::cout << C[i][j] << ' ';

}

std::cout << '\n';

}

return 0;

}

# **Результаты выполнения программы:**

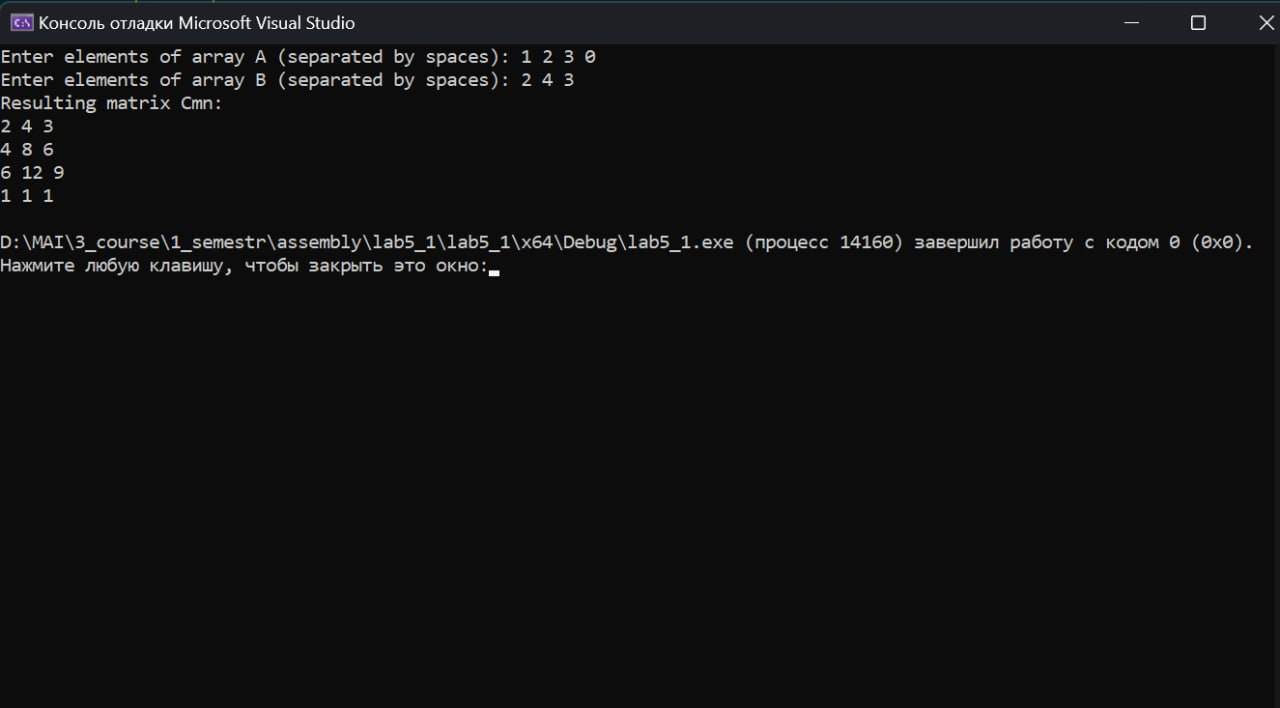
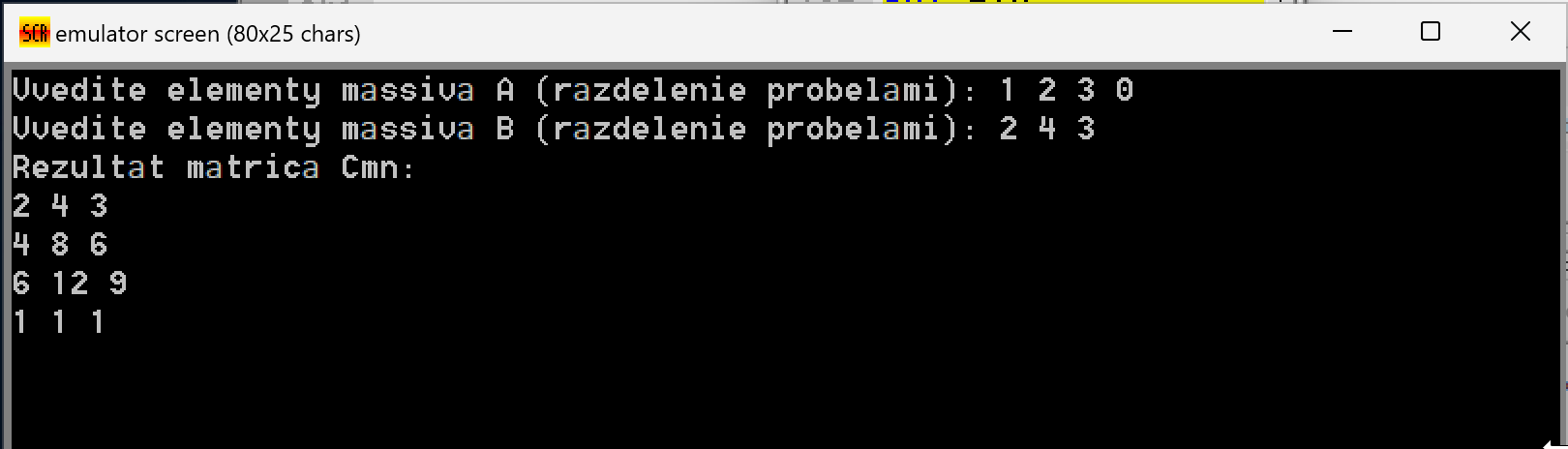


Рис. 1 — Результат выполнения программы на С++

Рис. 2 — Результат выполнения программы на Ассемблере