ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Московский институт электроники и математики им. А.Н. Тихонова  
Департамент компьютерной инженерии

Курс: Вычислительные системы и компьютерные сети

Отчёт   
по практической работе №4 «Адресация элементов массива, цикл, процедуры, стек.»

Студент: Омаров Марат Тимурович  
Группа: БИВ206  
Вариант: 8  
Дата: 13.10.2022

Москва 2022

Оглавление

[Условие задачи 3](#_Toc116594198)

[Код Intel 3](#_Toc116594199)

[Разбор инструкции 5](#_Toc116594200)

[Результат выполнения программы 6](#_Toc116594201)

[Код MIPS 6](#_Toc116594202)

[Результат выполнения программы 9](#_Toc116594203)

# Условие задачи

Для заданного двумерного массива А найти элементы, равные разнице двух соседних элементов по диагонали А(i, j)=А(i-1, j+1)-А(i+1, j-1). Вывести сообщение с числом таких элементов и адресом строки для последнего (с начала массива) встреченного элемента с указанными характеристиками или об их отсутствии.

# Код Intel

; Для заданного двумерного массива А найти элементы, равные

; разности двух соседних элементов по диагонали А(i, j)=А(i-1, j+1)-А(i+1, j-1).

; Вывести сообщение с числом таких элементов и адресом

; предпоследнего (с начала массива) встреченного элемента с указанными

; характеристиками или об их отсутствии.

format PE64 Console 5.0

entry Start

include 'win64a.inc'

section '.idata' import data readable

library kernel,'KERNEL32.DLL', msvcrt, 'msvcrt.dll'

import kernel, SetConsoleTitleA, 'SetConsoleTitleA',\

GetStdHandle, 'GetStdHandle', WriteConsoleA, 'WriteConsoleA',\

ReadConsoleA, 'ReadConsoleA', ExitProcess, 'ExitProcess'

import msvcrt, printf, 'printf',\

setlocale, 'setlocale',\

getch, '\_getch'

section '.data' data readable writeable

A dd 1, 1, 10, 7, 4,\

1, 4, 3, 4, 10,\

6, 4, 8, 9, 10,\

66, 1, 1, 55, 13

k\_str dd 4

k\_colum dd 5

k dd 0

adds dd 0

true\_coinc dd 0

dop dd 4

ru db 'Russian', 0

mes1 db 'Число элементов, удовлетворяющих условию: k = %d', 0dh, 0ah, 0

mes2 db 'Адрес предпоследнего элемента: %0x%016x', 0dh, 0ah, 0

mes\_error db 'Размеры строк и столбцов массива должны быть больше 2!!', 0dh, 0ah, 0

mes\_true db 'Предпоследнее число подходит по условию', 0dh, 0ah, 0

mes\_false db 'Предпоследнее число не подходит по условию', 0dh, 0ah, 0

section '.text' code readable executable

Proc:

pop rbp

; обращение к элементу массива через [A + №(строки)\*размер + №(элемента)\*размер]

pop r8 ; r8d = A

mov ecx, [k\_str]

sub ecx, 2 ; cx = k\_str - 2

cmp ecx, 0

jle incorrect\_input

mov edx, [k\_colum]

cmp edx, 2

jle incorrect\_input

mov ebx, 4 ; для того, чтобы начинать цикл со второй строки массива

for\_str:

mov eax, ebx

mul [k\_colum]

mov edx, [k\_colum]

sub edx, 2

add eax, r8d ; теперь eax - начало N строки массива

mov esi, 0

for\_colum:

inc esi

jmp calculations

continue:

dec edx

cmp edx, 0

jg for\_colum

add ebx, 4

loop for\_str

mov r12d, eax

xchg eax, esi

mul [dop]

add r12d, eax

mov [adds], r12d

invoke printf, mes2, [adds]

JMP exit\_proc

calculations:

mov r9, [eax+esi\*4]

mov r10d, eax

sub r10d, 20

mov r10d, [r10d+(esi+1)\*4] ; A[i-1, j+1]

mov r11d, eax

add r11d, 20

mov r11d, [r11d+(esi-1)\*4] ; A[i+1, j-1]

sub r10d, r11d

cmp r9d, r10d

jz coincidence

jmp continue

coincidence:

inc [k]

cmp edx, 1

jz last\_but\_one

jmp continue

last\_but\_one:

inc [true\_coinc]

jmp continue

incorrect\_input:

invoke printf, mes\_error

exit\_proc:

push rbp

RET

Start:

invoke setlocale, 0, ru

push A

call Proc

cmp [true\_coinc], 0

jg true\_mes

invoke printf, mes\_false

jmp Exit

true\_mes:

invoke printf, mes\_true

Exit:

invoke printf, mes1, [k]

invoke getch

invoke ExitProcess, 0

## Разбор инструкции

Разберём одну из инструкций адресации элемента массива в ОП. Выберем следующую:

mov r10d, [r10d+(esi+1)\*4] ; A[i-1, j+1]

В данной инструкции в регистр r10d помещается содержимое элемента массива, хранящееся по адресу r10d + (esi+1)\*4.

* r10d содержит адрес адрес начала “i-1” строки, ранее он был получен из суммы адреса начала массива, произведения номера строки на размер строки (i\*k\_colum\*4);
* esi определяет номер элемента внутри строки, т.е. он определяет номер от 0 до k\_colum – 1. В данном случае нам необходимо обратиться на элемент дальше, поэтому мы прибавляем единицу к номеру esi, а затем уже умножаем на размер элемента массива (4 байта, т.к. двойное слово).

## Результат выполнения программы

В соответствии с полученными флагами наша программа перейдёт в секцию “Less” с помощью условного перехода “jc Less” и выведется соответствующее сообщение о результате (рис. 1).

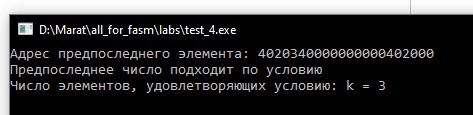


Рисунок 1. Результат выполнения программы Intel.

# Код MIPS

# Для заданного двумерного массива А найти элементы, равные разнице

# двух соседних элементов по диагонали А(i, j)=А(i-1, j+1)-А(i+1, j-1).

# Вывести сообщение с числом таких элементов и адресом строки для

# последнего (с начала массива) встреченного элемента с указанными

# характеристиками или об их отсутствии.

.data

A: .word 1 1 10 7 4

1 4 3 4 10

6 4 8 9 10

66 1 1 55 13

k\_str: .word 4

k\_colum: .word 5

k: .word 0

adds: .word 0

true\_coinc: .word 0

mes\_k: .asciiz "The number of elements that satisfy the condition: k = "

mes\_adds: .asciiz "\nThe address of penultimate element: "

.text

addi $sp, $sp, -12

la $a0, A

sw $a0, 8($sp)

la $a0, k\_str

sw $a0, 4($sp)

la $a0, k\_colum

sw $a0, 0($sp)

jal proc

#xor $t1, $t1, $t1

#lw $t1, 0($sp)

#sw $t1, k

#xor $t2, $t2, $t2

#lw $t2, 4($sp)

#sw $t2, true\_coinc

#xor $t3, $t3, $t3

#lw $t3, 8($sp)

#sw $t3, adds

addi $sp, $sp, 8

j exit

#---------------- начало процедуры -----------------#

proc:

lw $t0, 8($sp)

lw $t1, 4($sp)

lw $t2, 0($sp)

addi $sp, $sp, 12

addi $sp, $sp, -8

lw $t1, 0($t1) # кол-во строк

lw $t2, 0($t2) # кол-во столбцов

la $t9, 0

sub $t1, $t1, 2 # помещаем в t1 число строк, по которым нужно пройтись

la $t3, 1 # счётчик для прохода по строкам в цикле

loop\_str:

bltz $t1, exit\_proc

la $t4, 1 # счётчик для прохода по столбцам, начинаем со второго элемента

sub $t5, $t2, 2 # помещаем в t5 число столбцов, по которым нужно пройтись

loop\_colum:

sub $t5, $t5, 1

bltz $t5, continue\_str

j calculations

continue\_colum:

sub $t7, $t7, $t8

bne $t6, $t7, not\_equal

add $t9, $t9, 1

# теперь определим, предпоследний это элемент в строке или нет

la $t7, 0

la $t7, k\_str

lw $t7, 0($t7)

add $t7, $t7, $t2

sub $t7, $t7, 4 # храним в t7 значение k\_str + k\_colum - 4

la $t8, 0

add $t8, $t3, $t4 # храним в t8 число проходов цикла по строкам и столбцам

bne $t7, $t8, not\_equal # если они не равны, то элемент не препоследний

la $t7, 1

sw $t7, true\_coinc

#sw $t7, 4($sp)

not\_equal:

add $t4, $t4, 1

j loop\_colum

continue\_str:

sub $t1, $t1, 1

add $t3, $t3, 1

j loop\_str

calculations:

la $t6, 0

mul $t6, $t3, $t2

add $t6, $t6, $t4

mul $t6, $t6, 4

add $t6, $t6, $t0

xor $t7, $t7, $t7

move $t7, $t6

lw $t6, ($t6)

sub $t7, $t7, 4

sw $t7, adds # записали в переменную adds адрес

# (на последней итерации в переменную запишется адрес предпоследнего элемента)

#sw $t7, 8($sp)

la $t7, 0

sub $t7, $t3, 1

mul $t7, $t7, $t2

add $t7, $t7, $t4

add $t7, $t7, 1

mul $t7, $t7, 4

add $t7, $t7, $t0 # получили адрес A[i-1, j+1]

lw $t7, ($t7) # получили A[i-1, j+1]

la $t8, 0

add $t8, $t3, 1

mul $t8, $t8, $t2

add $t8, $t8, $t4

sub $t8, $t8, 1

mul $t8, $t8, 4

add $t8, $t8, $t0 # получили адрес A[i+1, j-1]]

lw $t8, ($t8) # получили A[i+1, j-1]

j continue\_colum

exit\_proc:

#sw $t9, 0($sp)

jr $ra

#----------- конец процедуры и начало вывода результатов ------------#

exit:

la $a0, mes\_k

li $v0, 4

syscall

move $a0, $t9

li $v0, 1

syscall

la $a0, mes\_adds

li $v0, 4

syscall

lw $a0, adds

li $v0, 1

syscall

li $v0, 10

syscall

## Результат выполнения программы

После выполнения программного кода выведутся следующие сообщения (см. рис. 2).

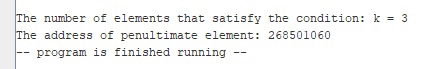


Рисунок 2. Результат выполнения программы MIPS.