

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ
по лабораторной работе №3
по дисциплине «Организация ЭВМ и систем»
ТЕМА: Представление и обработка целых чисел. Организация ветвящихся
процессов

Студент гр. 9383

Гордон Д.А.

Преподаватель

Ефремов М.А.

Санкт-Петербург

2020

Цель работы.

Разработать программу на языке ассемблер по организации ветвящихся процессов.

Текст задания.

Разработать на языке Ассемблера программу, которая по заданным целочисленным значениям параметров a , b , i , k вычисляет:

а) значения функций $i1 = f1(a,b,i)$ и $i2 = f2(a,b,i)$;

б) значения результирующей функции $res = f3(i1,i2,k)$,

где вид функций $f1$ и $f2$ определяется из табл. 2, а функции $f3$ - из табл.3 по цифрам шифра индивидуального задания ($n1,n2,n3$), приведенным в табл.4.

Значения a , b , i , k являются исходными данными, которые должны выбираться студентом самостоятельно и задаваться в процессе исполнения программы в режиме отладки. При этом следует рассмотреть всевозможные комбинации параметров a , b и k , позволяющие проверить различные маршруты выполнения программы, а также различные знаки параметров a и b .

Таблица 2	Таблица 3
$f1 = \begin{cases} / 15-2*i, & \text{при } a>b \\ \backslash 3*i+4, & \text{при } a\leq b \end{cases}$	$f1 = \begin{cases} / \min(i1,i2), & \text{при } k=0 \\ \backslash \max(i1,i2), & \text{при } k\neq 0 \end{cases}$
$f2 = \begin{cases} / -(4*i+3), & \text{при } a>b \\ \backslash 6*i-10, & \text{при } a\leq b \end{cases}$	$f2 = \begin{cases} / \max(i1,10-i2), & \text{при } k<0 \\ \backslash i1-i2 , & \text{при } k\geq 0 \end{cases}$
$f3 = \begin{cases} / 7-4*i, & \text{при } a>b \\ \backslash 8-6*i, & \text{при } a\leq b \end{cases}$	$f3 = \begin{cases} / i1+i2 , & \text{при } k=0 \\ \backslash \min(i1,i2), & \text{при } k\neq 0 \end{cases}$
$f4 = \begin{cases} / -(6*i-4), & \text{при } a>b \\ \backslash 3*(i+2), & \text{при } a\leq b \end{cases}$	$f4 = \begin{cases} / \min(i1-i2 , 2), & \text{при } k<0 \\ \backslash \max(-6, -i2), & \text{при } k\geq 0 \end{cases}$
$f5 = \begin{cases} / 20-4*i, & \text{при } a>b \\ \backslash -(6*i-6), & \text{при } a\leq b \end{cases}$	$f5 = \begin{cases} / \min(i1 , 6), & \text{при } k=0 \\ \backslash i1 + i2 , & \text{при } k\neq 0 \end{cases}$
$f6 = \begin{cases} / 2*(i+1)-4, & \text{при } a>b \\ \backslash 5-3*(i+1), & \text{при } a\leq b \end{cases}$	$f6 = \begin{cases} / i1-i2 , & \text{при } k<0 \\ \backslash \max(7, i2), & \text{при } k\geq 0 \end{cases}$
$f7 = \begin{cases} / -(4*i-5), & \text{при } a>b \\ \backslash 10-3*i, & \text{при } a\leq b \end{cases}$	$f7 = \begin{cases} / i1 + i2 , & \text{при } k<0 \\ \backslash \max(6, i1), & \text{при } k\geq 0 \end{cases}$
$f8 = \begin{cases} / -(6*i+8), & \text{при } a>b \\ \backslash 9-3*(i-1), & \text{при } a\leq b \end{cases}$	$f8 = \begin{cases} / i1 - i2 , & \text{при } k<0 \\ \backslash \max(4, i2 -3), & \text{при } k\geq 0 \end{cases}$

Таблица 4

№ студента	Шифр задания	№ студента	Шифр задания
1	1.2.1	14	3.4.2
2	1.3.2	15	3.5.3
3	1.4.3	16	3.6.4
4	1.5.4	17	3.7.5
5	1.6.5	18	3.8.6
6	1.7.6	19	4.5.7
7	1.8.7	20	4.6.8
8	2.3.8	21	4.7.2
9	2.4.7	22	4.8.3
10	2.5.6	23	5.6.4
11	2.6.5	24	5.7.5
12	2.7.4	25	5.8.6
13	2.8.3	26	6.8.1

Шифр задания – 1, 4, 3

$$f1 = \begin{cases} / 15-2*i, \text{ при } a>b \\ \backslash 3*i+4, \text{ при } a\leq b \end{cases}$$

$$f4 = \begin{cases} / -(6*i - 4), \text{ при } a>b \\ \backslash 3*(i+2), \text{ при } a\leq b \end{cases}$$

$$f3 = \begin{cases} / |i1 + i2|, \text{ при } k=0 \\ \backslash \min(i1,i2), \text{ при } k\neq 0 \end{cases}$$

Тестирование.

a	b	i	k	F1	F4	F3
3	4	1	0	7	9	16
4	3	-1	-1	17	10	10

Выводы

Была разработана программа на языке ассемблер, которая с помощью условных переходов высчитывала значения функций.

ПРИЛОЖЕНИЕ

КОД ПРИЛОЖЕНИЯ

L3.asm:

```
EOFLine EQU '$'      ; Определение символьной константы
a EQU 2
b EQU 1
I EQU 3
K EQU 0               ; "Конец строки"
```

; Стек программы

```
AStack SEGMENT STACK
        DW 12 DUP(?) ; Отводится 12 слов памяти
AStack ENDS
```

; Данные программы

```
DATA SEGMENT
```

; Директивы описания данных

```
F1 DW 0
F2 DW 0
F3 DW 0
DATA ENDS
```

; Код программы

```
CODE SEGMENT
        ASSUME CS:CODE, DS:DATA, SS:AStack
```

; Головная процедура

```
Main PROC FAR
        push DS      ;\ Сохранение адреса начала PSP в стеке
        sub AX,AX     ;> для последующего восстановления по
        push AX      ;/команде ret, завершающей процедуру
```

;записываем условие для F1

```
MOV AX,a      ; Заносим значение a
SUB AX,b      ; Вычитаем из a значение b
```

```

cmp ax,0      ; Сравниваем полученное а с 0
Jg Metka1     ; Переход к M1, если a>0
MOV AX,I      ; Заносим значение i
SHL AX,1      ; Сдвиг на 1б (умножаем на 2)
    MOV BX,I
ADD AX,BX     ;  $2*i + i = 3i$ 
;Mov BX,4     ;
ADD AX,4      ; Сложение ( $3*i + 4$ )
MOV F1,AX     ; Присваиваем F1 значение Ax
CMP F1,AX
JE Fun2
Metka1:       ;
mov AX,I      ;
SHL AX,1      ;  $2*i$ 
MOV BX,15
SUB BX,AX     ;  $15 - 2*i$ 
MOV F1,BX     ; Присваиваем F1 значение Bx

```

```

; записываем условие для F2
Fun2:
JG Metka2     ; Переход к M1, если a>0
mov AX,I      ; Заносим значение i
SHL AX,1      ; Сдвиг на 1б (умножаем на 2)
    MOV BX,I
ADD AX,BX     ;  $2*i + i = 3*i$ 
;MOV BX,6
ADD AX,6      ;  $3*(i + 2) = 3*i + 6$ 
MOV F2,Ax     ; Присваиваем F2 значение Ax
CMP F2,Ax     ; Сравнение результата
JE Fun3       ; Переход, если равно
Metka2:       ;
MOV AX,I      ;
SHL AX,1      ;  $2*i$ 
    SHL AX,1   ;  $2*i$ 
    MOV BX,I
ADD AX,BX     ;  $5*i$ 
    ADD AX,BX  ;  $6*i$ 
    MOV BX,4
SUB BX,AX     ;  $4 - 6*i$ 
MOV F2,BX     ; Присваиваем F2 значение Ax

```

```

; записываем условие для F3

```

```

Fun3:
MOV AX,K      ;
CMP AX,0      ;
JNE Metka3    ; Если k/=0, то переход
mov AX,F1     ;
mov BX,F2     ;
ADD BX,AX     ; i1 + i2
CMP BX,0      ;
JG Metka4     ;
                                ; i1 + i2 <= 0
xor BX, 0FFFFh ; инверсия всех битов
inc BX        ; добавляем 1
mov F3,BX     ;
RET          ;
Metka4:       ; i1 + i2 > 0
mov F3,BX     ; ответ записываем в F3
RET          ;
Metka3:       ;
MOV AX,F1     ;
MOV BX,F2     ;
SUB BX,AX     ;
CMP BX,0      ;
JG Metka5     ;
                                ; i2 <= i1
MOV F3,AX
ret
Metka5:       ;
MOV BX,F2     ; i2 > i1
MOV F3,BX     ;
ret          ; Выход в DOS по команде,
                                ; находящейся в 1-ом слове PSP.
CODE    ENDS
Main    ENDP
END main

```


L3.lst:

Microsoft (R) Macro Assembler Version 5.10

1/29/21 17:03:13

Page 1-1

```
= 0024          EOFLine EQU '$'      ; ОпределенИ
               ; символной константы
= 0002          a EQU 2
= 0001          B EQU 1
= 0003          I EQU 3
= 0000          K EQU 0              ; "КонИ
               ; Остроки"

               ; Стек программы

0000          AStack SEGMENT STACK
0000 000C[          DW 12 DUP(?)      ; Отводится
               12 слов памяти
               ????
               ]

0018          AStack ENDS

               ; Данные программы
0000          DATA SEGMENT

               ; Директивы описания данн
               ых

0000 0000          F1 DW 0
0002 0000          F2 DW 0
0004 0000          F3 DW 0
0006          DATA ENDS

               ; Код программы

0000          CODE SEGMENT
               ASSUME CS:CODE, DS:DATA, SS:AStack

               ; Головная процедура
0000          Main PROC FAR
0000 1E          push DS      ;\ Сохранени
               е адреса начала PSP в стеке
```

0001 2B C0	sub AX,AX ; > для послеИ
	восстановления по
0003 50	push AX ;/команде ret,
	завершающей процедуру
	;записываем условие
	для F1
0004 B8 0002	MOV AX,a ; Заносим знИ
	а
0007 2D 0001	SUB AX,b ; Вычитаем иИ
	а значение b
000A 3D 0000	cmp ax,0 ; Сравниваем
	полученное а с 0
000D 7F 18	Jg Metka1 ; Переход к М

```

                                1, если a>0
000F B8 0003                   MOV AX,I      ; Заносим знИ
                                ёR→□□i
0012 D1 E0                     SHL AX,1      ; Сдвиг на 1б
                                (умножаем на 2)
0014 BB 0003                   MOV BX,I
0017 03 C3                     ADD AX,BX      ; 2*i + i = 3i
0019 BB 0004                   Mov BX,4      ;
001C 03 C3                     ADD AX,BX      ; Сложение (3*
                                i + 4)
001E A3 0000 R                 MOV F1,AX     ; Присваивае
                                м F1 значение Ax
0021 39 06 0000 R              CMP F1,AX
0025 74 0E                     JE Fun2
0027                           Metka1:      ;
0027 B8 0003                   mov AX,I      ;
002A D1 E0                     SHL AX,1      ; 2*i
002C BB 000F                   MOV BX,15
002F 2B D8                     SUB BX,AX     ; 15 - 2*i
0031 89 1E 0000 R              MOV F1,BX     ; Присваивае
                                м F1 значение Bx

                                ; записываем условие
                                для F2
0035                           Fun2:
0035 7F 18                     JG Metka2     ; Переход к M
                                1, если a>0
0037 B8 0003                   mov AX,I      ; Заносим знИ
                                ёR→□□i
003A D1 E0                     SHL AX,1      ; Сдвиг на 1б
                                (умножаем на 2)
003C BB 0003                   MOV BX,I
003F 03 C3                     ADD AX,BX      ; 2*i + i = 3*i
0041 BB 0006                   MOV BX,6
0044 03 C3                     ADD AX,BX      ; 3*(i + 2) = 3*i + 6
0046 A3 0002 R                 MOV F2,Ax     ; Присваивае
                                м F2 значение Ax
0049 39 06 0002 R              CMP F2,Ax     ; Сравнение э

```

Решение

004D 74 17	JE Fun3 ; Переход, если равно
004F	Metka2: ;
004F B8 0003	MOV AX,I ;
0052 D1 E0	SHL AX,1 ; 2*i
0054 D1 E0	SHL AX,1 ; 2*i
0056 BB 0003	MOV BX,I
0059 03 C3	ADD AX,BX ; 5*i
005B 03 C3	ADD AX,BX ; 6*i
005D BB 0004	MOV BX,4
0060 2B D8	SUB BX,AX ; 4 - 6*i
0062 89 1E 0002 R	MOV F2,BX ; Присваиваем F2 значение Ax

```

; записываем условие
для F3

0066                               Fun3:
0066 B8 0000                       MOV AX,K      ;
0069 3D 0000                       CMP AX,0      ;
006C 75 1C                         JNE Metka3    ; Если к/=0, то
                                переход
006E A1 0000 R                     mov AX,F1     ;
0071 8B 1E 0002 R                   mov BX,F2     ;
0075 03 D8                         ADD BX,AX     ; i1 + i2
0077 83 FB 00                       CMP BX,0      ;
007A 7F 09                         JG Metka4     ;

                                ; i1 + i2 <= 0
007C 83 F3 FF                       xor BX, 0FFFFh ; инверсия вэ
                                8 битов
007F 43                             inc BX      ; добавл
                                яем 1
0080 89 1E 0004 R                   mov F3,BX     ;
0084 CB                             RET          ;
0085                               Metka4:         ; i1 + i2 > 0
0085 89 1E 0004 R                   mov F3,BX     ; ответ Й
                                8 битов
0089 CB                             RET          ;
008A                               Metka3:         ;
008A A1 0000 R                     MOV AX,F1     ;
008D 8B 1E 0002 R                   MOV BX,F2     ;
0091 2B D8                         SUB BX,AX     ;
0093 83 FB 00                       CMP BX,0      ;
0096 7F 04                         JG Metka5     ;
                                ; i2 <
                                = i1
0098 A3 0004 R                     MOV F3,AX
009B CB                             ret
009C                               Metka5:         ;
009C 8B 1E 0002 R                   MOV BX,F2     ; i2 > i1
00A0 89 1E 0004 R                   MOV F3,BX     ;
00A4 CB                             ret          ; Выход в DOS п

```

	о команде,
	; находящейс
	я в 1-ом слове PSP.
00A5	CODE ENDS
0000	Main ENDP
	END main

Segments and Groups:

N a m e	Length	Align	Combine	Class
ASTACK	0018	PARA		STACK
CODE	00A5	PARA		NONE
DATA	0006	PARA		NONE

Symbols:

N a m e	Type	Value	Attr	
A	NUMBER	0002		
B	NUMBER	0001		
EOFLINE	NUMBER	0024		
F1	L WORD	0000	DATA	
F2	L WORD	0002	DATA	
F3	L WORD	0004	DATA	
FUN2	L NEAR	0035	CODE	
FUN3	L NEAR	0066	CODE	
I	NUMBER	0003		
K	NUMBER	0000		
MAIN	F PROC	0000	CODE	Length = 0000
METKA1	L NEAR	0027	CODE	
METKA2	L NEAR	004F	CODE	
METKA3	L NEAR	008A	CODE	
METKA4	L NEAR	0085	CODE	
METKA5	L NEAR	009C	CODE	
@CPU	TEXT	0101h		
@FILENAME	TEXT	LR3		
@VERSION	TEXT	510		

117 Source Lines
117 Total Lines
24 Symbols

48072 + 455091 Bytes symbol space free

0 Warning Errors
0 Severe Errors