Министерство науки и высшего образования Российской Федерации
Калужский филиал
федерального государственного бюджетного
образовательного учреждения высшего образования
«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»
(КФ МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИУК «Информатика и управление»**

КАФЕДРА **ИУК4 «Программное обеспечение ЭВМ, информационные технологии»**

ЛАБОРАТОРНАЯ РАБОТА №4

«Использование команд условного перехода»

ДИСЦИПЛИНА: «Системное программирование»

Выполнил: студент гр. ИУК4-31Б _____ (Суриков Н. С.)
(подпись) (Ф.И.О.)

Проверил: _____ (Амеличева К. А.)
(подпись) (Ф.И.О.)

Дата сдачи (защиты):

Результаты сдачи (защиты):

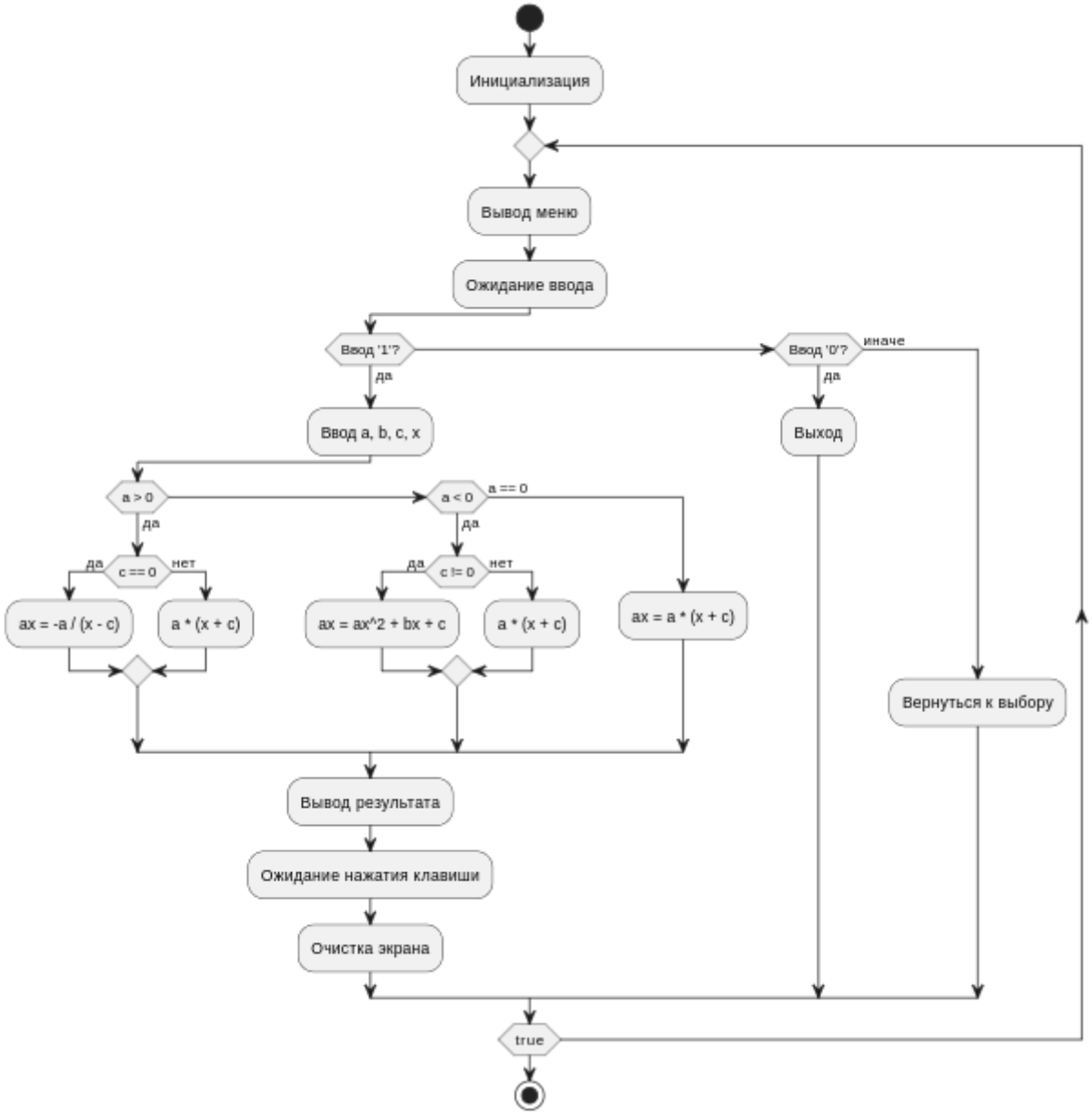
- Балльная оценка:
- Оценка:

Калуга, 2024

Цель: научиться выполнять операции с помощью команд условного перехода и безусловного перехода.

Вариант 3. $$F = \begin{cases} ax^2 + bx + c, & a < 0, c \neq 0 \\ \dfrac{-a}{x - c}, & a > 0, c = 0 \\ a(x + c), & \text{иначе} \end{cases}$$

Блок схема программы:



Листинг файла макросов:

```
  1    ; Макрос ввода 10-чного числа в регистр AX
  2  mReadAX macro buffer, sizee
  3                      local input, startOfConvert, endOfConvert
  4                      push  bx                                    ; Данные в стек
  5                      push  cx
  6                      push  dx
  7      input:
  8                      mov   [buffer], sizee                       ; Задаём размер буфера
  9                      mov   dx, offset [buffer]                   ; Поместить в регистр dx строку по
адресу buffer
 10                      mov   ah, 0Ah                               ; Чтение строки из консоли
 11                      int   21h                                   ; Прерывание DOS
 12
 13                      mov   ah, 02h                               ; Вывод символа на экран
 14                      mov   dl, 0Dh                               ; Перевод каретки на новую строку
 15                      int   21h                                   ; Прерывание DOS
 16
 17                      mov   ah, 02h                               ; Вывод символа на экран
 18                      mov   dl, 0Ah                               ; Чтение строки из консоли
 19                      int   21h                                   ; Прерывание DOS
 20
 21                      xor   ah, ah                                ; Очистка регистра ah
 22                      cmp   al, [buffer][1]                       ; Проверка на пустую строку
 23                      jz    input                                 ; Переход, если строка пустая
 24
 25                      xor   cx, cx                                ; Очистка регистра cx
 26                      mov   cl, [buffer][1]                       ; инициализация переменной-
счётчика
 27
 28                      xor   ax, ax                                ; Очистка регистра ax
 29                      xor   bx, bx                                ; Очистка регистра bx
 30                      xor   dx, dx                                ; Очистка регистра dx
 31
 32                      mov   bx, offset [buffer][2]                ; Поместить начало строки в
регистр bx
 33                      cmp   [buffer][2], '-'                      ; Проверка на знак числа
 34                      jne   startOfConvert                        ; Переход, если число
неотрицательное
 35                      inc   bx                                    ; Инкремент регистра bx
 36                      dec   cl                                    ; Декремент регистра-счетчика cl
 37      startOfConvert:
 38                      mov   dx, 10                                ; Поместить в регистр ax число 10
 39                      mul   dx                                    ; Умножение на 10 перед сложением
с младшим разрядом
 40                      cmp   ax, 8000h                             ; Проверка числа на выход за
границы
 41                      jae   input                                 ; Переход, если число выходит за
границы
 42                      mov   dl, [bx]                              ; Поместить в регистр dl следующий
символ
 43                      sub   dl, '0'                               ; Перевод его в числовой формат
 44                      add   ax, dx                                ; Прибавляем его к конечному
результату
 45                      cmp   ax, 8000h                             ; Проверка числа на выход за
границы
 46                      jae   input                                 ; Переход, если число выходит за
границы
 47                      inc   bx                                    ; Переход к следующему символу
 48                      loop  startOfConvert                        ; Цикл
 49                      cmp   [buffer][2], '-'                      ; Проверка на знак числа
 50                      jne   endOfConvert                          ; Переход, если число
неотрицательное
 51                      neg   ax                                    ; Инвертирование числа
 52      endOfConvert:
 53                      pop   dx                                    ; Данные из стека
 54                      pop   cx
```

```
                pop   bx
endm

; Макрос вывода 10-чного числа из регистра AX
mWriteAX macro
            local convert, write
            push  ax                ; Данные в стек
            push  bx
            push  cx
            push  dx
            push  di

            mov   cx, 10            ; cx - основание системы счисления
            xor   di, di            ; di - количество цифр в числе
            or    ax, ax            ; Проверка числа на ноль
            jns   convert           ; Переход, если число положительное
            push  ax                ; Регистр ax в стек
            mov   dx, '-'           ; Поместить в регистр dx символ '-'
            mov   ah, 02h           ; Вывод символа на экран
            int   21h               ; Прерывание DOS
            pop   ax                ; Регистр ax из стека
            neg   ax                ; Инвертирование отрицательного числа
    convert:
            xor   dx, dx            ; Очистка регистра dx
            div   cx                ; После деления dl = остатку от деления ax на cx
            add   dl, '0'           ; Перевод в символьный формат
            inc   di                ; Увеличение количества цифр в числе на 1
            push  dx                ; Регистр dx в стек
            or    ax, ax            ; Проверка числа на ноль
            jnz   convert           ; Переход, если число не равно нулю
    write:
            pop   dx                ; dl = очередной символ
            mov   ah, 02h           ; Вывод символа на экран
            int   21h               ; Прерывание DOS
            dec   di                ; Повторение, пока di != 0
            jnz   write

            pop   di                ; Данные из стека
            pop   dx
            pop   cx
            pop   bx
            pop   ax
endm

; Макрос вывода строки
mWriteStr macro string
             push ax
             push dx

             mov  ah, 09h
             mov  dx, offset string
             int  21h

             pop  dx
             pop  ax
endm

mClear macro                ;Макрос очистки экрана
          push ax
          push bx
          push cx
          push dx

          mov  ax,0600h     ; Подготавливает код для очистки экрана (функция 0).
          mov  bh, 4Ch      ; Устанавливает цвет фона и шрифта
          mov  cx, 0000     ; Указывает количество строк для очистки (все).
          mov  dx, 184FH    ; Указывает адрес экрана (184FH — адрес видеопамяти).
          int  10h          ; Вызывает прерывание BIOS для выполнения очистки экрана.

          pop  dx
```

```
            pop  cx
            pop  bx
            pop  ax
ENDM

mSetCursor MACRO row, col        ;Макрос установки курсора
               push ax
               push bx
               push cx
               push dx

               mov  ah, 02     ; Установка курсора
               mov  dh, row    ; номер строки в DH
               mov  dl, col    ; номер столбца в DL
               mov  bh, 0      ; Указывает страницу экрана (0).
               int  10h

               pop  dx
               pop  cx
               pop  bx
               pop  ax
ENDM
```

Листинг программы:

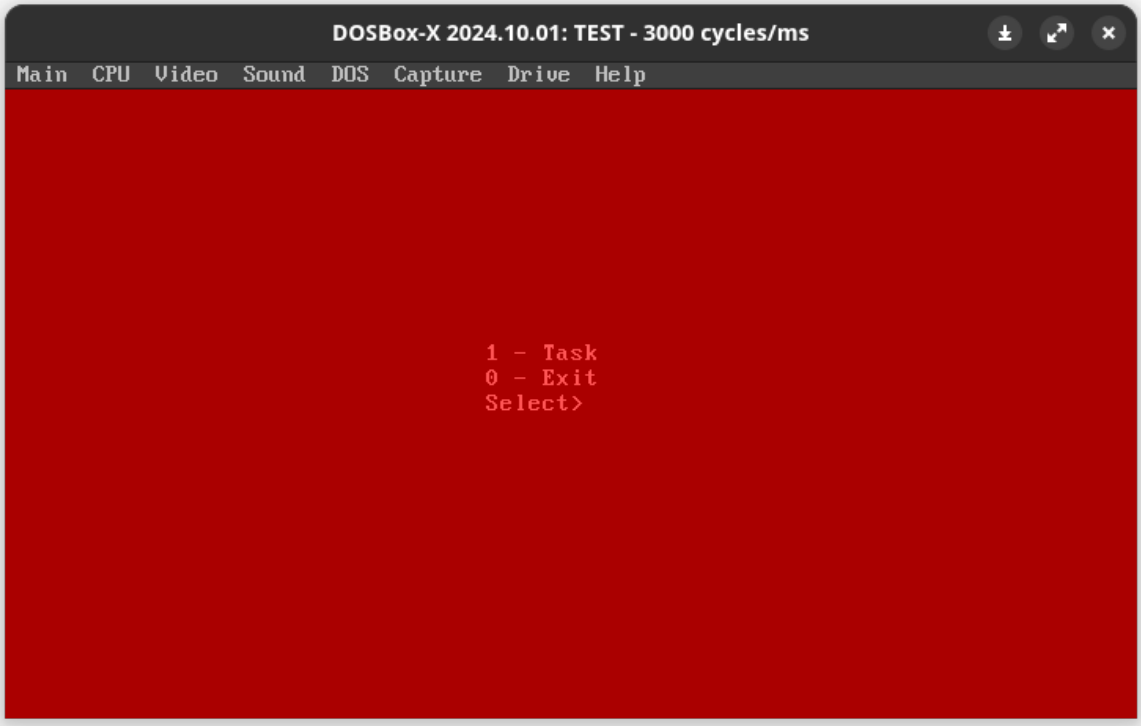
```
include macroses.asm
.model small
.stack 100h
.data
  buffer db 5, 0, 5 dup(0)
  mes_a  db 'Enter the number a: ', '$'
  mes_b  db 'Enter the number b: ', '$'
  mes_c  db 'Enter the number c: ', '$'
  mes_x  db 'Enter the number x: ', '$'
  a      dw 0
  b      dw 0
  c      dw 0
  x      dw 0
  menu1  db '1 - Task', '$',13,10
  menu2  db '0 - Exit','$',13,10
  select db 'Select> $'
  result db 'Result> $'
  task   db 'Your Task:$'
.code
  start:
                  mov        ax,@data
                  mov        ds,ax
                  mClear
  select_loop:
                  mSetCursor 10, 34
                  mWriteStr  menu1
                  mSetCursor 11, 34
                  mWriteStr  menu2
                  mSetCursor 12, 34
                  mWriteStr  select

                  mov        ah,01h
```

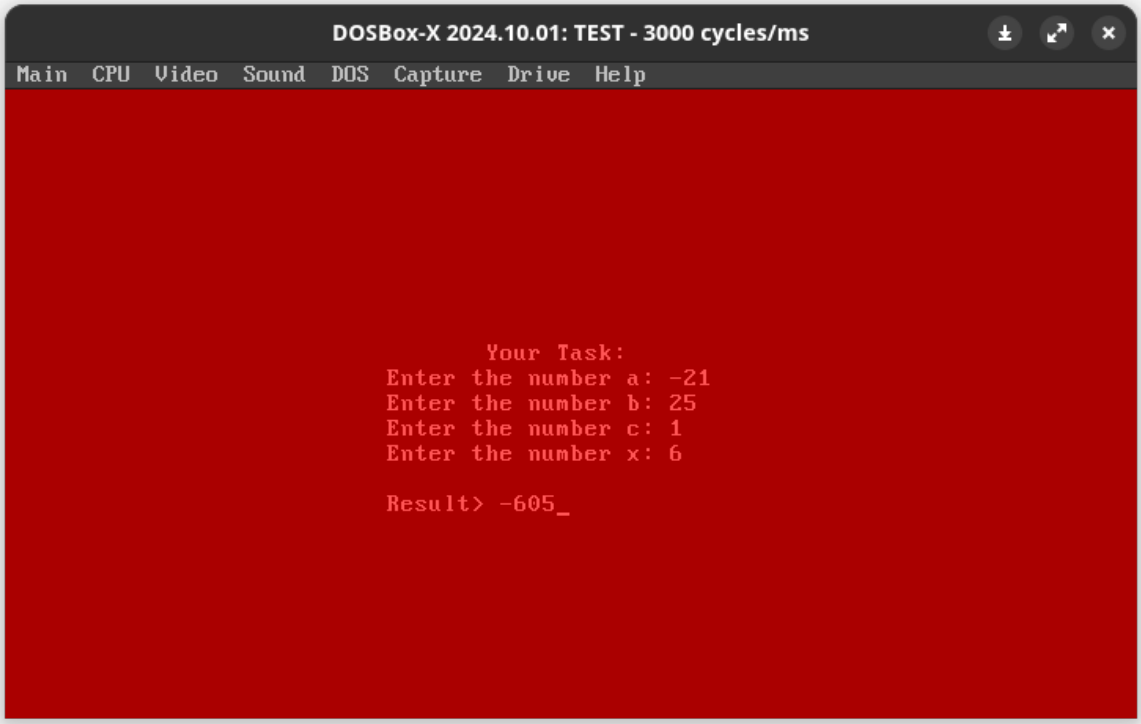
```
33                  int         21h
34
35                  cmp         al,'1'
36                  je          c1
37
38                  cmp         al,'0'
39                  je          exit
40
41                  jmp         select_loop
42  exit:
43                  mov         ah, 7h          ; Подготавливает функцию для ожидания нажатия
клавиши.
44                  int         21h             ; Вызывает прерывание DOS для ожидания нажатия
клавиши
45
46                  mov         ax,4c00h
47                  int         21h
48  c1:
49                  mClear
50                  mSetCursor 10, 34
51                  mWriteStr  task
52  ; Ввод переменной a
53                  xor         ax, ax
54                  mSetCursor 11, 27
55                  mWriteStr  mes_a
56                  mReadAX    buffer, 5
57                  mov         a, ax
58  ; Ввод переменной b
59                  xor         ax, ax
60                  mSetCursor 12, 27
61                  mWriteStr  mes_b
62                  mReadAX    buffer, 5
63                  mov         b, ax
64  ; Ввод переменной c
65                  xor         ax, ax
66                  mSetCursor 13, 27
67                  mWriteStr  mes_c
68                  mReadAX    buffer, 5
69                  mov         c, ax
70  ; Ввод переменной x
71                  xor         ax, ax
72                  mSetCursor 14, 27
73                  mWriteStr  mes_x
74                  mReadAX    buffer, 5
75                  mov         x, ax
76
77                  xor         ax,ax
78
79                  mov         ax,a
80                  cmp         ax,0
81                  jg          case_pozitive_a
82                  jl          case_negative_a
83                  jmp         else_case
84  case_pozitive_a:                            ; -a / (x-c)
```

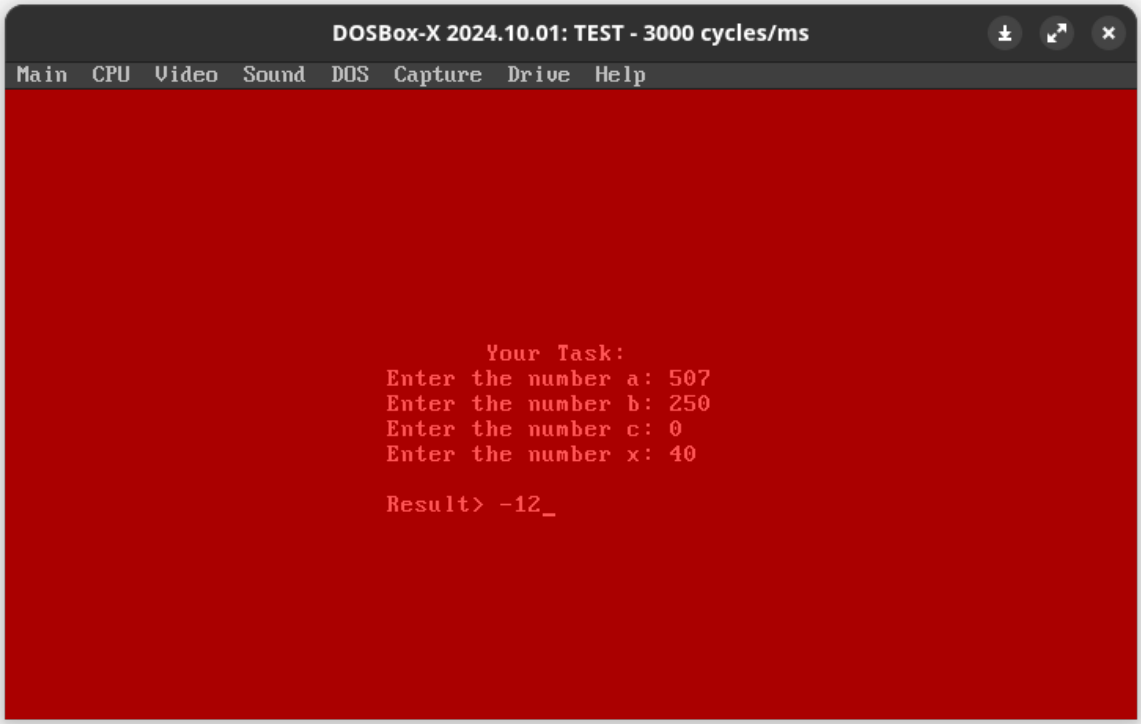
```
85
86                  xor         ax, ax
87                  mov         ax, c
88                  cmp         ax, 0
89                  jne         else_case
90
91                  xor         ax,ax
92                  xor         bx,bx
93                  xor         cx,cx
94                  xor         dx,dx
95
96                  mov         ax, a
97                  neg         ax
98
99                  mov         bx, x
100                 sub         bx, c
101
102                 cwd
103                 idiv        bx
104
105                 jmp         otvet
106 case_negative_a:                         ; ax^2 + bx + c
107                 xor         cx, cx
108                 mov         cx, c
109                 jcxz        else_case
110
111                 xor         ax,ax
112                 xor         bx,bx
113                 xor         cx, cx
114
115 ; Вычисляем x^2
116                 mov         ax, x           ; AX = x
117                 imul        ax              ; AX = x * x (x^2)
118                 mov         bx, ax          ; BX = x^2
119
120 ; Вычисляем ax^2
121                 mov         ax, a           ; AX = a
122                 imul        bx              ; AX = a * x^2
123                 mov         cx, ax          ; CX = a * x^2
124
125 ; Вычисляем bx
126                 mov         ax, b           ; AX = b
127                 imul        x               ; AX = b * x
128                 add         ax, cx          ; CX += b * x
129
130 ; Добавляем c
131                 add         ax, c           ; CX += c
132
133                 jmp         otvet
134 else_case:                                 ; a * (x + c)
135                 xor         ax,ax
136
137                 mov         ax, x
138                 add         ax, c
```

```
139                     imul        a
140
141                     jmp         otvet
142
143     otvet:
144                     mSetCursor 16, 27
145                     mWriteStr  result
146                     mWriteAX
147
148                     mov         ah, 7h          ; Подготавливает функцию для ожидания нажатия
клавиши.
149                     int         21h             ; Вызывает прерывание DOS для ожидания нажатия
клавиши
150
151                     mClear
152                     jmp         select_loop
153   end start
```

Результат выполнения программы:







Вывод: в ходе работы были получены навыки по использовать команд условного и безусловного перехода.