МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

ЛАБОРАТОРНАЯ РАБОТА №8

Отчёт о практике

студента 2 курса 251 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Тюменцева Радомира Александровича

Проверено:

Старший преподаватель _____ Е. М. Черноусова

Саратов 2025

1 Задание 8

Вариант: 6103

F = CONST op1 A op2 B

- CONST = 1Fh
- op1 = «-», op2 = «+»
- A, B - слова со знаком
- min(A, B)

1.1 Текст программы с комментариями

```
1  extrn GetStdHandle: proc,
2      lstrlenA: proc,
3      WriteConsoleA: proc,
4      ReadConsoleA: proc,
5      ExitProcess: proc
6
7  .data
8  ; Макрозамены для номеров стандартных потоков
9  STD_OUTPUT_HANDLE equ -11; Номер стандартного потока вывода
10 STD_INPUT_HANDLE  equ -10; Номер стандартного потока ввода
11
12 ; Глобальные переменные
13 hStdInput  dq ? ; Дескриптор стандартного потока ввода
14 hStdOutput dq ? ; Дескриптор стандартного потока вывода
15 f_value    dq ? ; Переменная для хранения результата вычисления F
16 min_value  dq ? ; Переменная для хранения минимального значения из A и B
17
18 ; Строки пользовательского интерфейса
19 a_prompt             db 'a = ', 0
20 b_prompt             db 'b = ', 0
21 f_message            db 'F = 1Fh - A + B = ', 0
22 min_message          db 'min(A, B) = ', 0
23 invalid_char_message db 'Invalid character', 0
24 out_of_range_message db 'Value is out of range', 0
25 exit_message         db 'Press any key to exit...', 0
26 new_line             db 0Dh, 0Ah, 0
27
28 .code
29 ; Макросы для работы со стеком
30 ; Макрос для выравнивания стека по 16-байтовой границе и выделения места
   для аргументов процедур
31 STACKALLOC macro arg
32   push R15
33   mov  R15, RSP     ; Сохранение текущего указателя стека (RSP) в R15
34   sub  RSP, 8 * 4   ; Выделение места в стеке для 4-х аргументов (RCX,
     RDX, R8, R9)
35   if arg            ; Если макрос вызван с аргументом,
```

```
36      sub RSP, 8 * arg; то выделяется дополнительное место в стеке
37    endif
38    and  SPL, 0F0h    ; Выравнивание стека (младший байт, SPL) по 16-
      байтовой границе
39  endm
40
41  ; Макрос для восстановления стека в исходное состояние
42  STACKFREE macro
43    mov RSP, R15
44    pop R15
45  endm
46
47  ; Макрос для обнуления пятого аргумента в стеке
48  ; Предназначен для функций ReadConsoleA и WriteConsoleA
49  NULL_FIFTH_ARG macro
50    mov qword ptr[RSP + 32], 0
51  endm
52
53  ; Процедура вывода строки в консоль
54  PrintString proc uses RAX RCX RDX R8 R9 R10 R11, string: qword
55    local bytesWritten: qword; Число записанных байт
56    STACKALLOC 1              ; Выделение места в стеке под 5 аргументов
57    mov RCX, string           ; Указатель на строку для определения её длины
58    call lstrlenA             ; Определение длины строки
59    mov RCX, hStdOutput       ; Дескриптор потока вывода
60    mov RDX, string           ; Указатель на строку для вывода
61    mov R8, RAX               ; Длина строки
62    lea R9, bytesWritten      ; Указатель на число выведенных байт
63    NULL_FIFTH_ARG
64    call WriteConsoleA        ; Вывод строки
65    STACKFREE
66    ret 8                     ; Возврат и очистка стека от одного qword
      аргумента
67  PrintString endp
68
69  ; Процедура чтения знакового числа из консоли
70  ReadNumber proc uses RBX RCX RDX R8 R9
71    local readStr[64]: byte, bytesRead: dword; Буфер для строки и количество
      прочитанных байт
72    STACKALLOC 2       ; Выделение места в стеке под 6 аргументов
73    mov RCX, hStdInput; hConsoleInput: дескриптор потока ввода
74    lea RDX, readStr   ; lpBuffer: адрес буфера для записи строки
75    mov R8, 64         ; nNumberOfCharsToRead: максимальный размер буфера
76    lea R9, bytesRead ; lpNumberOfCharsRead: адрес переменной для реальной
      длины строки
77    NULL_FIFTH_ARG     ; lpInputControl: для ANSI-строк должен быть 0
78    call ReadConsoleA
79
80    ; Парсинг строки в число
81    xor RCX, RCX
```

```
 82   mov ECX, bytesRead; Реальная длина строки
 83   sub ECX, 2         ; Удаление символов переноса строки и возврата каретки
 84   xor RBX, RBX       ; RBX будет накапливать результат (число)
 85   mov R8, 1          ; R8 будет использоваться для умножения каждой цифры
      на 10
 86
 87   ; Обработка строки по символам
 88   loopString:
 89     dec RCX             ; Переход к предыдущему символу
 90     cmp RCX, -1         ; Проверка на конец строки
 91     je scanningComplete ; Если дошли до начала строки, цикл завершается
 92     xor RAX, RAX
 93     mov AL, readStr[RCX]; Выбор текущего символа
 94     cmp AL, '-'         ; Проверка на знак минус
 95     jne eval
 96     neg RBX             ; Если вначале стоит минус, то меняем знак числа
 97     cmp RCX, 0          ; Проверка, что минус стоит в начале
 98     je scanningComplete ; Если да, цикл завершается
 99     jmp error           ; Иначе, выводится ошибка
100
101     ; Проверка, является ли символ десятичной цифрой
102     eval:
103       cmp AL, 30h ; Сравнение символа с кодом '0'
104       jl error    ; Если символ меньше '0', выводится ошибка
105       cmp AL, 39h ; Сравнение символа с кодом '9'
106       jg error    ; Если символ больше '9', выводится ошибка
107       sub RAX, 30h; Вычитание кода '0' для преобразования в число
108       mul R8      ; Умножение на разряд
109       add RBX, RAX; Добавление к итоговому числу в RBX
110       ; Обновление множителя для следующего разряда
111       mov RAX, 10
112       mul R8
113       mov R8, RAX
114       jmp loopString
115
116     error:
117       mov R10, 1; Установка флага ошибки
118       STACKFREE
119       ret
120
121     scanningComplete:
122       mov R10, 0
123       mov RAX, RBX; Возвращение числа в RAX
124       STACKFREE
125       ret
126 ReadNumber endp
127
128 ; Процедура вывода числа в консоль
129 PrintNumber proc uses RAX RCX RDX R8 R9 R10 R11, number: qword
130   local numberStr[22]: byte; Локальный буфер для строки
```

```
131   STACKALLOC 1              ; Выделение места в стеке
132   xor R8, R8                ; R8 - счетчик/индекс для строки numberStr
133   mov RAX, number           ; Перемещение числа для вывода в RAX
134   cmp number, 0             ; Проверка знака числа
135   jge positive
136   ; Если число отрицательное
137   mov numberStr[R8], '-'    ; Добавление знака "минус" в начало строки
138   inc R8                    ; Увеличение индекса
139   neg RAX                   ; Инвертирование числа для работы с
      положительным значением
140
141   positive:
142     mov RBX, 10 ; Делитель для разделения числа на цифры
143     xor RCX, RCX; Счетчик количества цифр
144
145     ; Разделение числа на цифры
146     divToDigits:
147       xor RDX, RDX; Обнуление RDX (остатка от предыдущего деления)
148       div RBX     ; RDX:RAX = RAX / RBX
149       add RDX, 30h; Преобразование цифры в ASCII-код
150       push RDX    ; Добавление цифры в стек
151       inc RCX     ; Увеличение счетчика цифр
152       cmp RAX, 0  ; Проверка на окончание деления
153       jne divToDigits
154
155     ; Сборка строки из цифр
156     createString:
157       pop RDX               ; Получение ASCII-кода цифры из стека
158       mov numberStr[R8], DL; Добавление цифры в строку
159       inc R8                ; Увеличение индекса
160       loop createString
161
162     mov numberStr[R8], 0; Добавление нуля в конец строки
163     lea RAX, numberStr  ; Перемещение адреса строки в RAX
164     push RAX
165     call PrintString    ; Вывод строки
166     ret 8               ; Возврат и очистка стека от одного qword
        аргумента
167 PrintNumber endp
168
169 ; Процедура ожидания ввода
170 WaitEnter proc uses RAX RCX RDX R8 R9 R10 R11
171   local readStr:byte, bytesRead:dword
172   STACKALLOC 1
173
174   ; Вывод сообщения 'Press any key to exit...'
175   mov RCX, hStdOutput
176   lea RDX, exit_message
177   mov R8D, 25; Длина строки 'Press any key to exit...'
178   lea R9, bytesRead
```

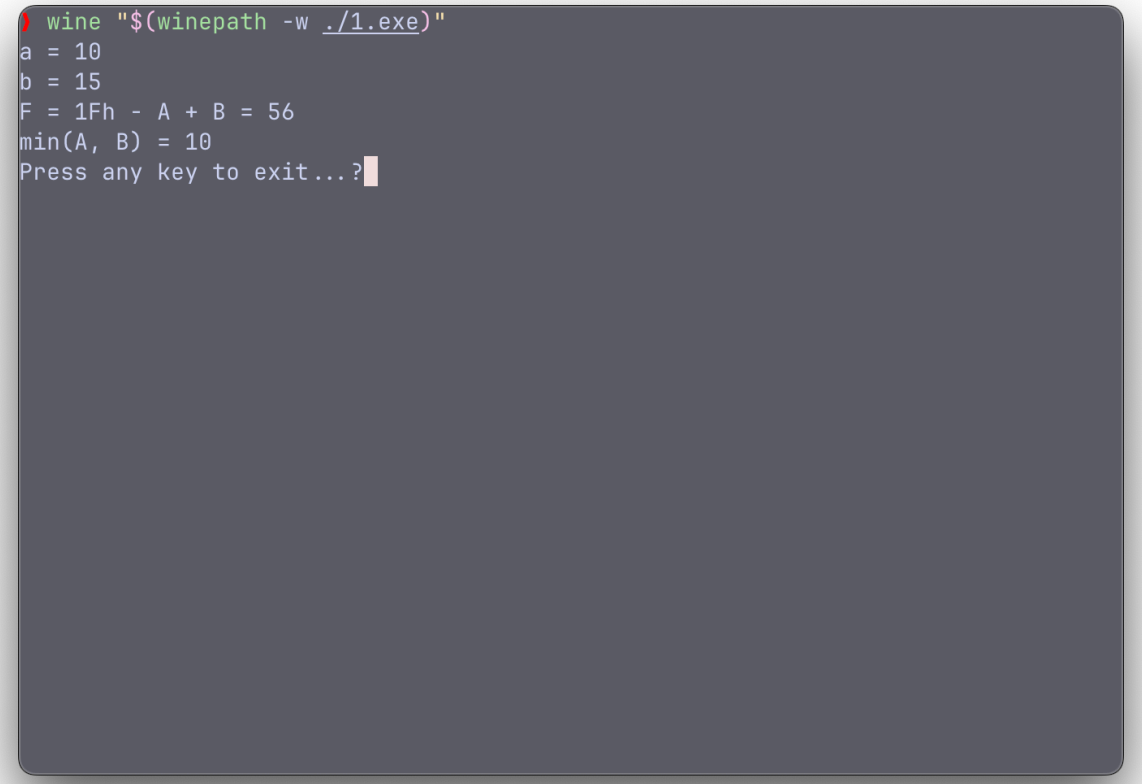
```
179   NULL_FIFTH_ARG
180   call WriteConsoleA
181
182   ; Ожидание ввода одного символа
183   mov RCX, hStdInput
184   lea RDX, readStr
185   mov R8D, 1
186   lea R9, bytesRead
187   NULL_FIFTH_ARG
188   call ReadConsoleA
189
190   STACKFREE
191   ret
192 WaitEnter endp
193
194 ; Процедура вывода переноса строки
195 PrintNewLine proc
196   lea RAX, new_line
197   push RAX
198   call PrintString
199   ret
200 PrintNewLine endp
201
202 ; Главная процедура
203 mainCRTStartup proc
204   STACKALLOC 0
205
206   ; Инициализация дескрипторов консоли
207   mov RCX, STD_OUTPUT_HANDLE
208   call GetStdHandle
209   mov hStdOutput, RAX
210
211   mov RCX, STD_INPUT_HANDLE
212   call GetStdHandle
213   mov hStdInput, RAX
214
215   ; Ввод и проверка числа A
216   lea RAX, a_prompt        ; Вывод текста "a = "
217   push RAX
218   call PrintString
219
220   call ReadNumber          ; Чтение числа с консоли
221   cmp R10, 1               ; Проверка флага ошибки из ReadNumber
222   je invalid_char_exception; Если ошибка, переход к завершению
223   ; Проверка на переполнение для знакового 16-битного числа (word)
224   cmp RAX, 32767
225   jg out_of_range_exception
226   cmp RAX, -32768
227   jl out_of_range_exception
228   mov R8, RAX              ; Сохранение первого числа в R8
```

```
229
230   ; Ввод и проверка числа B
231   lea RAX, b_prompt         ; Вывод текста "b = "
232   push RAX
233   call PrintString
234
235   call ReadNumber           ; Чтение числа с консоли
236   cmp R10, 1                ; Проверка флага ошибки
237   je invalid_char_exception; Если ошибка, переход к завершению
238   ; Проверка на переполнение для знакового 16-битного числа (word)
239   cmp RAX, 32767
240   jg out_of_range_exception
241   cmp RAX, -32768
242   jl out_of_range_exception
243   mov R9, RAX               ; Сохранение второго числа в R9
244
245   ; Поиск min(A, B)
246   cmp R8, R9
247   jg R9_is_min              ; Если A > B, то B - минимум
248   mov min_value, R8         ; Иначе A - минимум
249   jmp R8_is_min
250   R9_is_min:
251   mov min_value, R9
252   R8_is_min:
253
254   ; Вычисление и вывод F = 1Fh - A + B
255   lea RAX, f_message; Выводим сообщение "F = ..."
256   push RAX
257   call PrintString
258
259   add R8, R9                ; R8 = A + B
260   add R8, 1Fh               ; R8 = A + B + 1Fh
261   mov f_value, R8
262   push f_value              ; Помещение результата в стек для вывода
263   call PrintNumber
264
265   call PrintNewLine
266
267   ; Вывод min(A, B)
268   lea RAX, min_message      ; Вывод сообщения "min(A, B) = "
269   push RAX
270   call PrintString
271
272   push min_value            ; Помещение минимального значения в стек для
      вывода
273   call PrintNumber
274
275   ; Переход к завершению программы
276   jmp exit_proc
277
```

```
278   ; Обработка исключений
279   invalid_char_exception:
280     lea RAX, invalid_char_message
281     push RAX
282     call PrintString
283     jmp exit_proc
284
285   out_of_range_exception:
286     lea RAX, out_of_range_message
287     push RAX
288     call PrintString
289
290   exit_proc:
291     call PrintNewLine
292     call WaitEnter; Ожидание нажатия любой клавиши
293     STACKFREE
294     xor RCX, RCX  ; Код завершения 0
295     call ExitProcess
296 mainCRTStartup endp
297
298 end
```

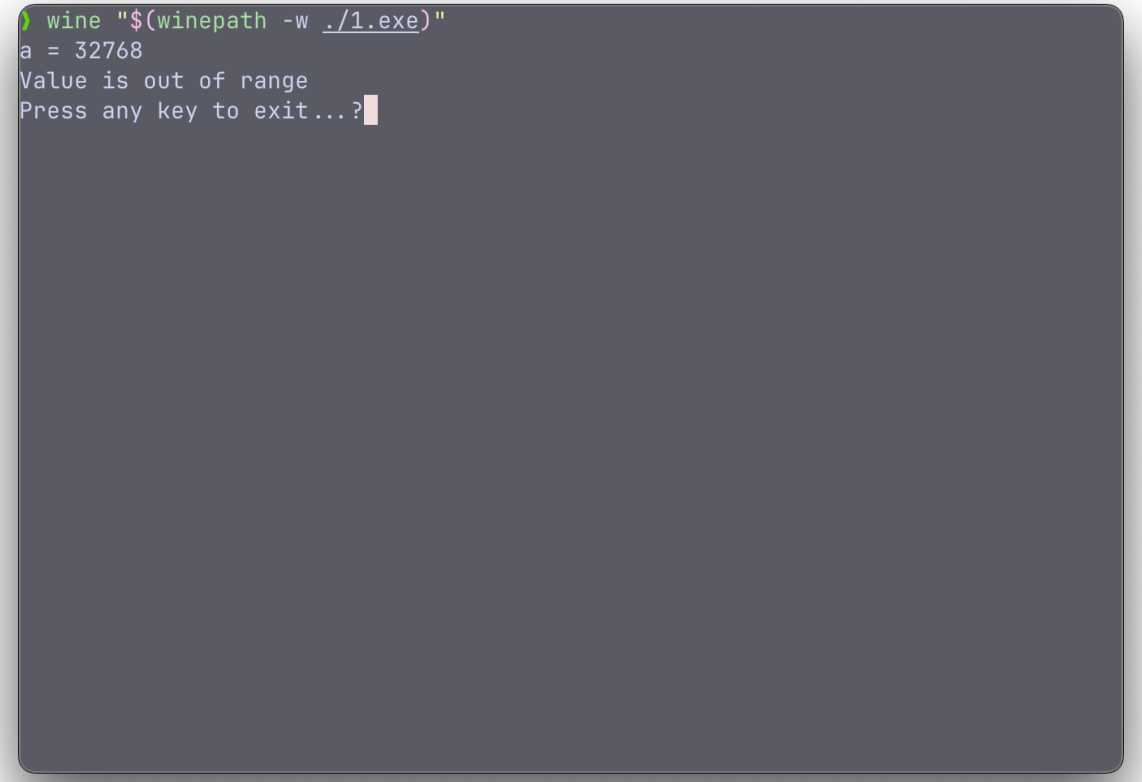
Текст программы

1.2 Скриншоты запуска программы

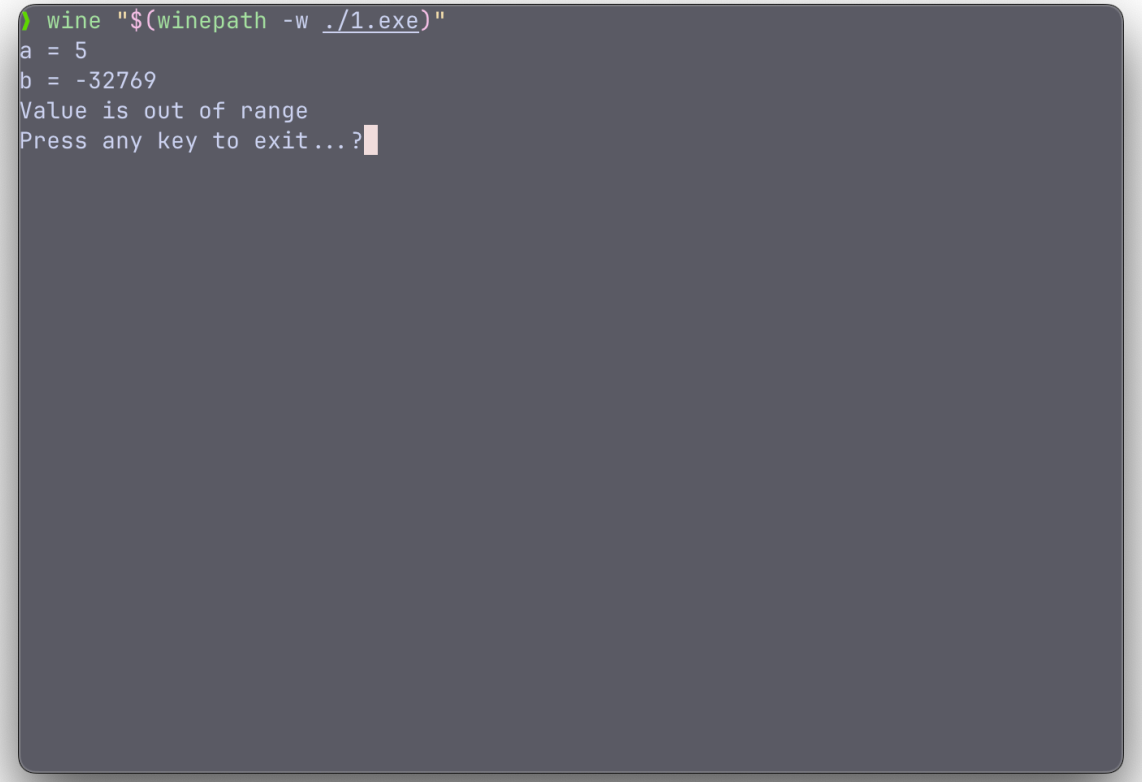


С положительным ответом

С отрицательным ответом



С вводимым числом A, которое выходит за границы возможных значений



С вводимым числом B, которое выходит за границы возможных значений

С параметром, содержащим кроме цифр другие символы

С отрицательным параметром