

Национальный исследовательский университет

«Высшая школа экономики»

Факультет компьютерных наук

Департамент

Программная инженерия

Микропроект по дисциплине

«Архитектура вычислительных систем»

Тема работы: Программа, вычисляющая с помощью степенного ряда с точностью не хуже 0,05% значение функции $\sqrt{1+x}$ для заданного параметра x (использовать FPU). Используемый язык программирования – Assembly (FASM).

Выполнил студент группы БПИ191 Бен Мустафа А.Р.

Преподаватель: Легалов Александр Иванович

Москва 2020

Вариант 5.

Разработать программу, вычисляющую с помощью степенного ряда с точностью не хуже 0,05% значение функции $\sqrt{1+x}$ для заданного параметра x (использовать FPU).

Работа программы организована следующим образом:

1. Получение аргумента из командной строки и его обработка (приведение строки к числу).
2. Проверка выполнения условия $|x| \leq 1$. Вывод указания на ошибку в случае некорректного ввода пользователя.
3. Вычисление значения степенного ряда
4. Предоставление полученного ответа пользователю.

Весь процесс работы программы разделён на процедуры, выполняющие те или иные функции.

В коде программы предоставлены поясняющие комментарии.

Ниже представлен полный код программы:

flat assembler 1.73.25

File Edit Search Run Options Help

```
format PE GUI 4.0
entry start
include 'win32ax.inc'

;Секция для хранения данных, доступна в программе только для чтения
section '.data' data readable
errmsg db 'Предупреждение!',0
hlpmsg db 'Обнаружена проблема в заданной командной строке!',13,10
       db 'Проверьте, что выполняется условие |x| <= 1 и программа запускается корректно!',13,10
       db 'Пример запуска из командной строки: start /Path/microproject.exe X, где X - переданный параметр для ряда',0

capt   db 'Сумма степенного ряда',0
fmt1   db '%lg',0
fmt     db 'x = %lg',13,10
        db 'Сумма степенного ряда: %lg',0
e       dd 0.0005
c1      dd 1.0

;Ниже - секция с кодом программ
section '.code' code readable executable
start:  ; Стартовая точка программы
        call main                ;вызов функции main
        invoke ExitProcess,0     ;выход

;Основная процедура программы
;Входные параметры отсутствуют, процедура ничего не возвращает
main:
        push ebp                 ;пролог функции
        mov ebp,esp              ;создание кадра стека
        sub esp,408h             ;создание локальных переменных
x       equ ebp-408h
z       equ ebp-400h             ;результатирующая строка
        push ebx                 ;сохранение регистров
        push esi
        push edi
        stdcall [GetCommandLine] ;принимаем командную строку
        mov edi,eax              ;получаем адрес командной строки
        ccall [strlen],eax        ;получаем длину командной строки
        mov ebx,eax
        cmp byte [edi],'"'       ;если строка начинаеся с кавычки
        jz quotes                ;совершаем переход
        . . .
```



File Edit Search Run Options Help

```
mov al, ' ' ; иначе имя запускаемого файла отделено пробелом
mov ecx, ebx
repne scasb ; ищем пробел или конец строки
jmp fnd ; продолжаем
quotes: mov al, '"' ; ищем две пары кавычек
mov ecx, ebx
repne scasb ; первую
repne scasb ; и вторую
fnd: lea eax, [x] ; получаем адрес переменной в стеке
ccall [sscanf], edi, fmt1, eax ; распознаем число
test eax, eax ; проверяем результат
jg calc

; В случае ошибки выдаем пользователю соответствующее уведомление
er: stdcall [MessageBox], 0, hlpmsg, errmsg, 0
jmp ex ; выход

; Вычисляем то, что нужно
calc: fld qword [x] ; x
fabs ; |x|
fcomp [c1] ; сравниваем |x| с 1 (по условию |x| < 1)
fstsw ax ; перенести флаги сравнения в ax
sahf ; занести ah в флаги процессора
ja er ; если |x| > 1, значит неправильный аргумент
fld [e] ; e
sub esp, 8 ; выделить в стеке место под double
fstp qword [esp] ; записать в стек double число
fld qword [x] ; x
sub esp, 8 ; выделить в стеке место под double
fstp qword [esp] ; записать в стек double число
call mysqrt ; Вычислить mysqrt(x, e)
add esp, 16 ; удалить переданные параметры

sub esp, 8 ; передать значение
fstp qword [esp] ; функции через стек
fld1 ; 1
fadd qword [x] ; 1+x
fsqrt ; вычисление точного значения sqrt(1+x)
sub esp, 8 ; передать значение
fstp qword [esp] ; функции через стек
fld qword [x] ; x
sub esp, 8 ; передать значение (x)
fstp qword [esp] ; функции через стек
push fmt ; формат сообщения
lea ebx, [s] ; адрес формируемого сообщения
```



File Edit Search Run Options Help

```
    push ebx
    call [sprintf]          ;сформировать результат
    add esp,32              ;коррекция стека
    invoke MessageBox,0,ebx,capt,MB_OK ;вывести результат
ex:  pop edi                ;восстановить регистры
    pop esi
    pop ebx
    leave
    ret                    ;выход из функции

;Процедура вычисления sqrt(a) с точностью epsilon
mysqrt:
    push ebp                ;создать кадр стека
    mov ebp,esp
    sub esp,20              ;создание локальных переменных
;локальные переменные
tmp  equ ebp-20
a    equ ebp-16             ;значение на текущем шаге
p    equ ebp-8              ;значение на предыдущем шаге
    fldl                    ;1
    fstp qword [a]          ;a=1
    xor ecx,ecx             ;n=0
    fldz                    ;s=0
lp:  fld qword [a]           ;a
    fst qword [p]           ;p=a
    faddp st1,st            ;s=s+a
    fld qword [a]           ;a
    mov eax,1
    sub eax,ecx
    sub eax,ecx             ;1-2*n
    mov [tmp],eax
    fimul dword [tmp]       ;a*(1-2*n)
    inc ecx                 ;n++
    lea eax,[ecx*2]         ;2n
    mov [tmp],eax
    fmul qword [ebp+8]      ;a*x
    fidiv dword [tmp]       ;a*x/(2n)
    fst qword [a]           ;сохранить a
    fsub qword [p]          ;a-p
    fabs                    ;|a-p|
    fcomp qword [ebp+16]    ;сравнить |a-p| с e
    fstsw ax                ;перенести флаги сравнения в ax
    sahf                    ;занести ah в флаги процессора
    jae lp                  ;если |a-p| >= e, продолжаем цикл
```



File Edit Search Run Options Help

```
lp:    fld qword [a]           ; a
       fst qword [p]         ; p=a
       faddp st1,st          ; s=s+a
       fld qword [a]         ; a
       mov eax,1
       sub eax,ecx
       sub eax,ecx           ; 1-2*n
       mov [tmp],eax
       fimul dword [tmp]     ; a*(1-2*n)
       inc ecx               ; n++
       lea eax,[ecx*2]       ; 2n
       mov [tmp],eax
       fmul qword [ebp+8]    ; a*x
       fidiv dword [tmp]     ; a*x/(2n)
       fst qword [a]         ; сохранить a
       fsub qword [p]        ; a-p
       fabs                  ; |a-p|
       fcomp qword [ebp+16]  ; сравнить |a-p| с e
       fstsw ax              ; перенести флаги сравнения в ax
       sahf                  ; занести ah в флаги процессора
       jae lp                ; если |a-p| >= e, продолжаем цикл
       fadd qword [a]        ; прибавить последнее слагаемое
       leave                  ; эпилог функции
       ret
```

```
section '.idata' import data readable writeable
```

```
library kernel,'KERNEL32.DLL',\
    msvcrt,'MSVCRT.DLL',\
    user32,'USER32.DLL'
```

```
import kernel,\
    strlen,'strlenA',\
    GetCommandLine,'GetCommandLineA',\
    ExitProcess,'ExitProcess'
```

```
import user32,\
    MessageBox,'MessageBoxA'
```

```
import msvcrt,\
    sprintf,'sprintf',\
    sscanf,'sscanf',\
    printf,'printf'
```