ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук Департамент

программной инженерии

Самостоятельная работа по дисциплине:

"Архитектура вычислительных систем"

Программа вычисления корня квадратного по итерационной формуле Герона Александрийского с точностью не хуже 0,05% (использовать FPU)

Исполнитель

Студент группы БПИ195

Кенесбек Ерасыл

Почта: ekenesbek@edu.hse.ru

Москва 2020

Задание:

Разработать программу вычисления корня квадратного по итерационной формуле Герона Александрийского с точностью не хуже 0,05% (использовать FPU).

Решение:

Итерационная формула Герона имеет вид

$$x_{n+1} = \frac{1}{2}(x_n + \frac{a}{x_n}),$$

где a — фиксированное положительное число, а x_1 — любое положительное число.

Итерационная формула задаёт убывающую (начиная со 2-го элемента) последовательность, которая при любом выборе x_1 быстро сходится к величине $\sqrt{a}$(квадратный корень из числа), то есть

$$\lim_{n\to\infty} x_n = \sqrt{a}$$

Эту формулу можно получить, применяя метод Ньютона к решению уравнения $a - x^2 = 0$.

Источник: Википедия

Пример:

Попробуем вычислить квадратный корень для 25, используя округления при вычислениях. Пусть нашим первым предположением для значения $\sqrt{25}$ будет значение 3.

n	x_n	$x_{n+1} = \frac{1}{2}(x_n + \frac{a}{x_n})$	Приблизительное значение x_{n+1}
1	3	$\frac{1}{2}(3 + \frac{25}{3})$	$\frac{1}{2}(3 + 8.33) = \frac{1}{2} \cdot 11.33 \approx 5.67$
2	5.67	$\frac{1}{2}(5.67 + \frac{25}{5.67})$	$\frac{1}{2}(5.67 + 4.41) = \frac{1}{2} \cdot 10.08 = 5.04$
3	5.04	$\frac{1}{2}(5.04 + \frac{25}{5.04})$	$\frac{1}{2}(5.04 + 4.96) = \frac{1}{2} \cdot 10 = 5$
4	5	$\frac{1}{2}(5 + \frac{25}{5})$	$\frac{1}{2}(5 + 5) = \frac{1}{2} \cdot 10 = 5$

Попробуем написать программу, которая вычисляет значение функции гиперболического тангенса с помощью степенного ряда на привычном нам языке C#:

```
class Program
{
	static void Main(string[] args)
	{
		double a = 2.5;
		double xp;
		double e = 0.00005;
		double x = a;
		do
		{
			xp = x;
			x = (x + a / x) / 2;
		}while (Math.Abs(x - xp) >= e);
		Console.WriteLine(x);
		Console.ReadLine();

	}
}
```

Цикл завершится, когда `Math.Abs(x - xp) >= e` очередной член суммы по модулю будет меньше заданной точности . Точность 0,05% совпадает с абсолютной величиной 0,0005.

Теперь перепишем все на FASM:

Для вывода результата воспользуемся системной функцией int MessageBox(

```
    HWND hWnd,
    LPCTSTR lpText,
    LPCTSTR lpCaption,
    UINT uType
);
```

Которая создаёт окно сообщения. Она имеет следующие аргументы:

- hWnd — дескриптор родительского окна;
- lpText — указатель на строку с сообщением;
- lpCaption — указатель на строку с заголовком окна;
- uType — 32битное значение, определяющее содержимое окна.

Чтобы создать строку с сообщением применим функцию sprintf, которая форматирует данные по заданному шаблону.

int sprintf (char * str, const char * format, ...);

Она имеет следующие аргументы:

- str — указатель на символьный буфер с результатом;
- format — указатель на C-строку, содержащую формат результата;
- остальные аргументы — данные, подлежащие форматированию.

Так как у данной функции количество аргументов не постоянно, то она использует соглашение вызова cdecl, которое отличается от stdcall тем, что очистку стека от аргументов выполняет вызывающая функция, а не вызываемая.

Для завершения работы программы выполним вызов функции ExitProcess, которая имеет один аргумент — код завершения работы программы.

VOID ExitProcess(

UINT uExitCode // код выхода для всех потоков

);

Для получения командной строки воспользуемся функцией GetCommandLine, которая не имеет аргументов и возвращает указатель на ASCIIZ командную строку. Первым параметром командной строки является полный путь к запускаемому файлу программы. В зависимости от расположения и наличия пробелов в имени, он может быть обернут символами «кавычки», это следует учитывать при разборе командной строки.

Длину ASCIIZ-строки можно определить при помощи функции size_t strlen(const char * string);

Параметром которой является указатель на строку, а результатом количество символов до завершающего нулевого байта.

Чтобы преобразовать строку в действительное значение, применим функцию sscanf, которая распознает и считывает данные по заданному шаблону из строки.

int sscanf(char *buf, const char *format, arg-list);

Она имеет следующие аргументы:

● buf — указатель на символьный буфер, подлежащий считыванию; ● format — указатель на С-строку, содержащую формат результата; ● остальные аргументы — данные, подлежащие форматированию.

Входной параметр a считывается из командной строки программы. Программу можно разбить на следующие функции:

main – главная функция программы. Выделяем ее для удобства использования вспомогательных локальных переменных. Функция входных

параметров не имеет и ничего не возвращает, вызывается по соглашению stdcall. Имеет локальные переменные:

s – byte[1024] - результирующая строка; a – qword –

входной параметр a из командной строки;

mysqrt – функция вычисления гиперболического синуса аргумента (первый параметр) путем разложения в степенной ряд с заданной точностью (второй параметр). Вызывается по соглашению cdecl. Имеет локальные переменные:

xp – значение x на предыдущем шаге Результат

возвращается в ST(0).

Код программы:

```
format PE GUI 4.0 entry
start
include 'include\win32ax.inc'
;Секция данных содержит строковые переменные и константы,
доступна только для чтения section '.data' data readable errmsg
db 'Ошибка командной строки',0
hlpmsg db 'Программа должна запускаться в формате: sqrt a',13,10
     db 'Где a - действительное положительное число',13,10
     db 'Пример: sqrt 2.5',0

capt db 'Вычисление квадратного корня по формуле Герона',0
fmt1 db '%lg',0 fmt db 'x = %lg',13,10  db 'Точное значние:
%lg',13,10     db 'Значение, полученное по итерационной
формуле: %lg',0 e dd 0.0005 ;точность 0.05%
c2 dd 2
;секция кода
section '.code' code readable executable
start:  ;начало  программы    call  main
;вызов функции main
     invoke ExitProcess,0     ;выход
;Главная функция программы
;Входных параметров нет (stdcall)
;Ничего не возвращает main:
     push ebp       ;пролог функции     mov
ebp,esp   ;создание кадра стека
     sub esp,408h        ;создание локальных переменных
```

```
;локальные переменные a
equ ebp-408h
s equ ebp-400h  ;результирующая строка  push ebx  ;сохранить
регистры по соглашению stdcall  push esi  push edi
     stdcall [GetCommandLine] ;получаем командную
строку    mov edi,eax    ;адрес командной строки
ccall [lstrlen],eax      ;длина командной строки
mov ebx,eax    ;длина командной строки
     cmp byte [edi],'"' ;если строка начинаеся с "
 jz quotes   ;то переход   mov al,' '   ;иначе имя запускаемого
файла отделено пробелом
     mov ecx,ebx    ;длина строки       repne scasb
;ищем пробел или конец строки      jmp fnd
;продолжить
quotes: mov al,'"'            ;ищем две пары кавычек
     mov ecx,ebx    ;длина строки
repne scasb    ;первую
     repne scasb         ;и вторую
fnd: lea eax,[a]    ;адрес переменой в стеке
ccall [sscanf],edi,fmt1,eax ;распознаем в число   test
eax,eax ;проверить результат
     jg calc
;если ошибка, вывести сообщение
er: stdcall [MessageBox],0,hlpmsg,errmsg,0 ;вывод ошибки
     jmp ex         ;выход
er1: fstp st   ;удалить число из сопроцессора     jmp
er   ;вывести сообщение об ошибке
calc:  fld  qword  [a]  ;a   ftst
;сравнить  a  с  0           fstsw
ax ;перенести флаги сравнения в
ax         sahf     ;занести  ah
в флаги процессора  jbe      er1
;если x<=0, значит неправильный
аргумент

     fld [e]        ;
     sub esp,8 ;выделить в стеке место под double
fstp qword [esp];записать в стек double число
sub esp,8 ;выделить в стеке место под double
fstp qword [esp];записать в стек double число
call mysqrt ;Вычислить mysqrt(x)
     add esp,16     ;удалить переданные параметры

     sub esp,8      ;передать значение (s)
     fstp qword [esp]        ;функции через стек
fld qword [a]       ;a   fsqrt
```

```
;вычисление точного значения       sub esp,8
;передать значение (sqrt(a))
     fstp qword [esp]        ;функции через
стек      fld qword [a]       ;a   sub esp,8
;передать значение (a)
     fstp qword [esp]        ;функции через стек
push fmt       ;формат сообщения   lea ebx,[s]
;адрес формируемого сообщения      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
     ;эпилог функции
     ret            ;выход из функции

;double mysqrt(double a,double eps)
;вычисление sqrt(a) с точностью eps
;соглашение вызова cdecl mysqrt:
     push ebp       ;создать кадр стека
mov ebp,esp
     sub esp,08h         ;создание локальных переменных
;локальные переменные
xp equ ebp-8h            ;значение x на предыдущем шаге
     fld qword [ebp+8]   ;x=a

lp: fst qword [xp]       ;xp=x
     fld qword [ebp+8]   ;a
     fld st1             ;x
     fdivp st1,st        ;a/x
     faddp st1,st        ;x+a/x
     fidiv [c2]          ;x=(x+a/x)/2
     fld st              ;x
     fsub qword [xp]          ;x-xp
     fabs           ;|x-xp|
     fcomp [e] ;сравнить |x-xp| с xp
        fstsw ax ;перенести флаги сравнения в ax
        sahf        ;занести ah в флаги процессора
     jae lp         ;если |x-xp| >= xp, продолжаем цикл

     leave              ;эпилог функции
     ret

section '.idata' import data readable writeable
library kernel,'KERNEL32.DLL',\
msvcrt,'MSVCRT.DLL',\    user32,'USER32.DLL'
```

```
  import kernel,\
         lstrlen,'lstrlenA',\
       GetCommandLine,'GetCommandLineA',\
         ExitProcess,'ExitProcess'

  import user32,\
     MessageBox,'MessageBoxA'

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

Результат выполнения программы см. рис. 1.

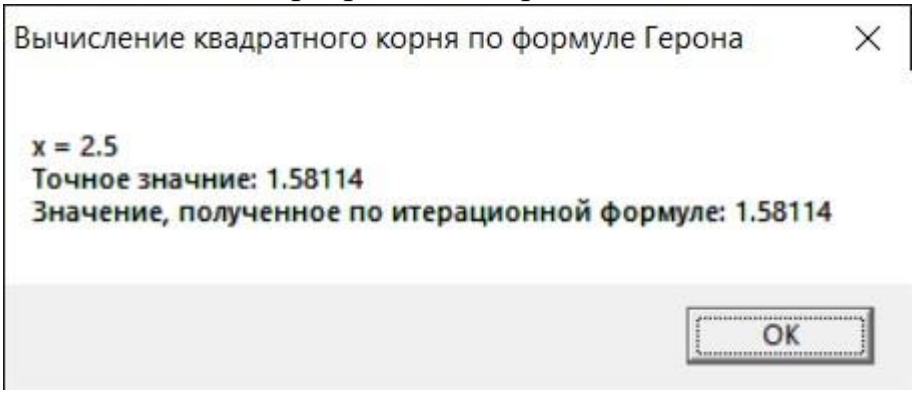

Рисунок 1. Результат выполнения программы

При отсутствия параметра в командной строке или неположительного параметра будет выведено сообщение следующее (см. рис. 2).

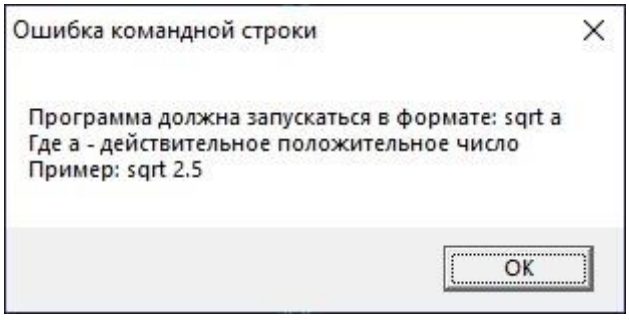

Рисунок 2. Результат запуска без параметра или с неположительным параметром