МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №5**по дисциплине: «Команды сопроцессора»

Выполнил: ст. группы ПВ-211

Чувилко Илья Романович

Проверил:  
Осипов Олег Васильевич

Белгород 2023 г.

**Цель работы:** изучение команд сопроцессора для выполнения арифметических операций.

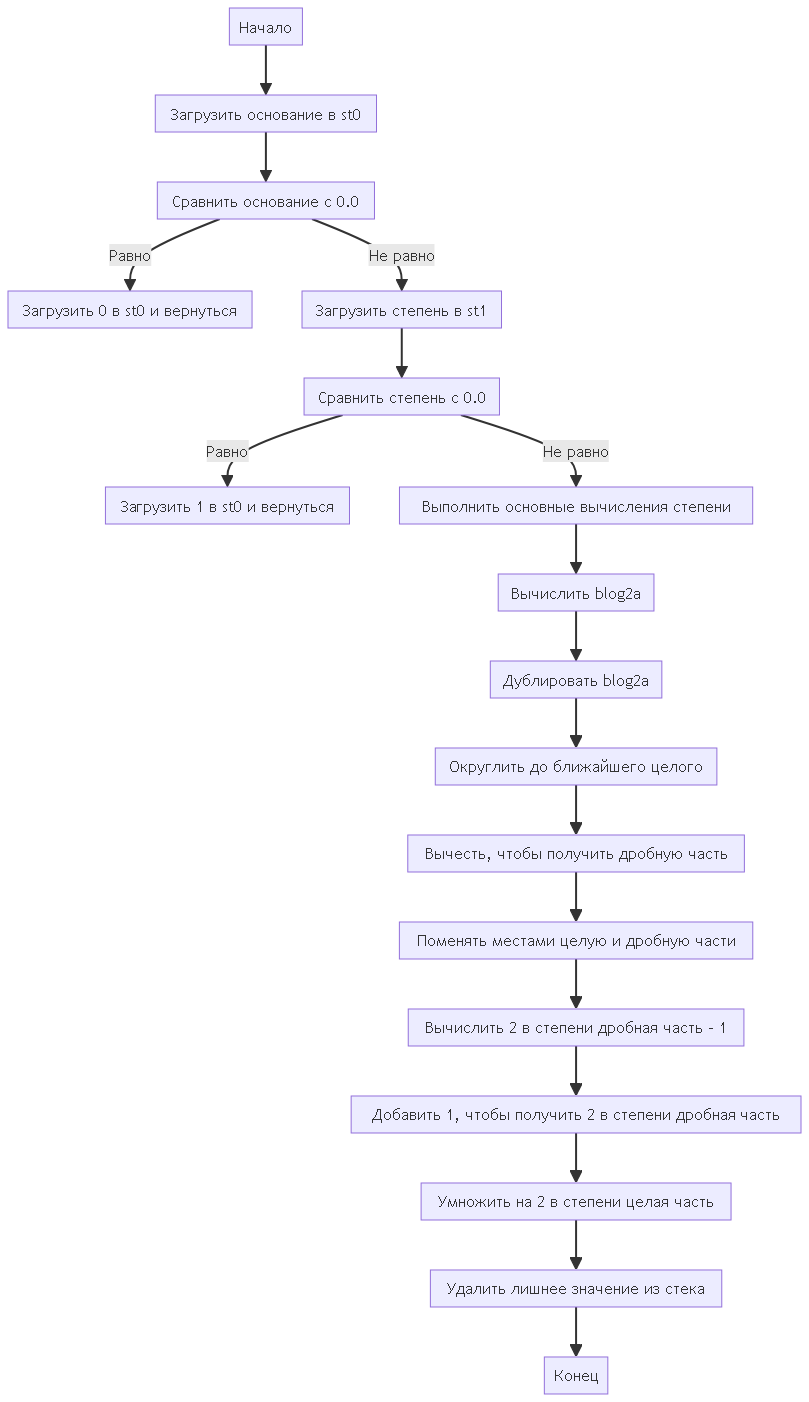
|  |  |
| --- | --- |
| 20 |  |

**Задания для выполнения к работе**

1. Написать функцию pow (x, y) для возведения числа x в степень y. Числа x, y могут быть произвольными, в том числе отрицательными. Рассмотреть случаи,когда x = 0 и/или y = 0. Аргументы передавать подпрограмме через стек. Если алгоритм требует выгрузки чисел из сопроцессора в память или регистры, использовать для этого стек. Подобрать набор тестовых данных для проверки работы функции pow (не менее 10). Убедиться в том, что результаты работы написанной функции pow и стандартной функции pow библиотеки math.h языка C или функции Math.Pow языка C# совпадают. В отчёт включить текст программы, блок-схему алгоритма функции pow и набор тестовых данных.
2. 2. Численно исследовать на сходимость ряд. Аргументы тригонометрических функций считать в радианах. Для возведения чисел в степень использовать написанную функцию pow. В отчёт включить текст программы и значения суммы ряда при n от 1 до 50. Вывести результат на экран в виде:
3. n = 1; S = …
4. n = 2; S = …
5. …

**Выполнение:**

**Задание 1**

****

**Код программы:**

; Аргументы: base, power

pow proc

; Проверяем, равен ли base 0

fld st(0) ; Копируем base

ftst ; Сравниваем с 0.0

fstsw ax ; Сохраняем состояние сопроцессора в ax

sahf ; Загружаем флаги состояния в флаги процессора

je base\_is\_zero

; Проверяем, равен ли power 0

fld st(1) ; Копируем power

ftst ; Сравниваем с 0.0

fstsw ax ; Сохраняем состояние сопроцессора в ax

sahf ; Загружаем флаги состояния в флаги процессора

je power\_is\_zero

fstp st(0)

fstp st(0)

; Основной код возведения в степень

fyl2x;

fld st(0);

frndint;

fsub st(1), st(0)

fxch st(1)

f2xm1

fld1

faddp st(1), st

fscale

fstp st(1)

ret

base\_is\_zero:

fld st(2)

ftst

fstsw ax

sahf

je both\_zero

fldz

fstp st(1)

jmp end\_pow

power\_is\_zero:

fld1

fstp st(1)

jmp end\_pow

both\_zero:

fld1

fstp st(1)

jmp end\_pow

end\_pow:

ret

pow endp

**Результаты тестов:**

| **Base** | **Power** | **Expected Result** |
| --- | --- | --- |
| 0.50 | 2.00 | 0.25 |
| 2.00 | 0.50 | 1.414214 |
| 1.50 | 1.50 | 1.837117 |
| -10.00 | 3.00 | -1000.0 |
| 2.00 | 10.00 | 1024.0 |
| 0.10 | 2.00 | 0.01 |
| 3.00 | 0.33 | 1.436978 |
| 5.00 | 2.50 | 55.901699 |
| 0.75 | 1.20 | 0.708066 |
| 6.00 | 0.00 | 1.0 |
| -0.50 | 2.00 | 0.25 |
| 0.00 | 0.00 | 1 |
| 0.00 | 2.00 | 0 |
| 2.00 | 0.00 | 1 |
| -0.50 | 0.00 | 1.0 |

**Задание 2**

.686

.MMX

.XMM

.MODEL Flat, StdCall

OPTION CaseMap:None

.stack 4096

include \masm32\include\windows.inc

include \masm32\include\masm32.inc

include \masm32\include\msvcrt.inc

include \masm32\include\kernel32.inc

includelib \masm32\lib\masm32.lib

includelib \masm32\lib\msvcrt.lib

includelib \masm32\lib\kernel32.lib

.DATA

message DB "Result: %f", 0

base DQ 2.0

power DQ 4.0

res DQ ?

.CODE

; Аргументы: base, power

pow proc

; Проверяем, равен ли base 0

fld st(0) ; Копируем base

ftst ; Сравниваем с 0.0

fstsw ax ; Сохраняем состояние сопроцессора в ax

sahf ; Загружаем флаги состояния в флаги процессора

je base\_is\_zero

; Проверяем, равен ли power 0

fld st(1) ; Копируем power

ftst ; Сравниваем с 0.0

fstsw ax ; Сохраняем состояние сопроцессора в ax

sahf ; Загружаем флаги состояния в флаги процессора

je power\_is\_zero

fstp st(0)

fstp st(0)

; Основной код возведения в степень

fyl2x;

fld st(0);

frndint;

fsub st(1), st(0)

fxch st(1)

f2xm1

fld1

faddp st(1), st

fscale

fstp st(1)

ret

base\_is\_zero:

fld st(2)

ftst

fstsw ax

sahf

je both\_zero

fldz

fstp st(1)

jmp end\_pow

power\_is\_zero:

fld1

fstp st(1)

jmp end\_pow

both\_zero:

fld1

fstp st(1)

jmp end\_pow

end\_pow:

ret

pow endp

start:

fld qword ptr [power]

fld qword ptr [base]

call pow

fstp qword ptr [res]

push dword ptr [res+4]

push dword ptr [res]

push offset [message]

call crt\_printf

add esp, 12

; Завершение программы

push 0

call ExitProcess

end start

**Результат работы программы:**

**n = 1; S = 2.142607**

**n = 2; S = 3.703267**

**n = 3; S = 4.861648**

**n = 4; S = 5.736648**

**n = 5; S = 6.408198**

**n = 6; S = 6.930946**

**n = 7; S = 7.342914**

**n = 8; S = 7.671039**

**n = 9; S = 7.934748**

**n = 10; S = 8.148299**

**n = 11; S = 8.322332**

**n = 12; S = 8.464910**

**n = 13; S = 8.582232**

**n = 14; S = 8.679123**

**n = 15; S = 8.759383**

**n = 16; S = 8.826034**

**n = 17; S = 8.881497**

**n = 18; S = 8.927731**

**n = 19; S = 8.966326**

**n = 20; S = 8.998583**

**n = 21; S = 9.025570**

**n = 22; S = 9.048166**

**n = 23; S = 9.067099**

**n = 24; S = 9.082972**

**n = 25; S = 9.096286**

**n = 26; S = 9.107458**

**n = 27; S = 9.116835**

**n = 28; S = 9.124709**

**n = 29; S = 9.131322**

**n = 30; S = 9.136877**

**n = 31; S = 9.141544**

**n = 32; S = 9.145466**

**n = 33; S = 9.148762**

**n = 34; S = 9.151531**

**n = 35; S = 9.153859**

**n = 36; S = 9.155816**

**n = 37; S = 9.157461**

**n = 38; S = 9.158844**

**n = 39; S = 9.160007**

**n = 40; S = 9.160985**

**n = 41; S = 9.161806**

**n = 42; S = 9.162498**

**n = 43; S = 9.163079**

**n = 44; S = 9.163567**

**n = 45; S = 9.163978**

**n = 46; S = 9.164323**

**n = 47; S = 9.164614**

**n = 48; S = 9.164858**

**n = 49; S = 9.165063**

**n = 50; S = 9.165236**

**Вывод:** в ходе лабораторной работы мы изучили команды сопроцессора для выполнения арифметических операций.