

Правительство Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования «Национальный

исследовательский университет «Высшая школа экономики»

Факультет компьютерных наук

Департамент программной инженерии

Отчет к домашнему заданию По дисциплине

«Архитектура вычислительных систем»

Работу выполнил:

Студент группы БПИ-195 Гуницкий Р.Я.

Москва 2020

Задача

Разработать программу, определяющую число чисел-палиндромов (в восьмеричном представлении) в диапазоне от 1 до 10^6

Решение

Число является палиндромом в восьмеричном представлении в том случае, если оно одинаково читается в обоих направлениях в восьмеричной системе счисления. Например число 349 является палиндромом в восьмеричном представлении так как при переводе его в восьмеричную систему счисления мы получим число 535, которое читается и в ту и в другую сторону одинаково.

Опишем словами алгоритм работы программы: нужно пройти циклом от 1 до 10^6 и проверить каждое число. Каждое число представляется в виде массива состоящего из чисел от 0 до 7. Этот массив является восьмеричной записью проверяемого числа. После получения восьмеричной записи числа нужно пройти по массиву с двух сторон проверяя противоположные числа на равенство. Таким образом получается, что алгоритм проходится с начала до середины массива и с конца до середины массива одновременно, сравнивая противоположные значения. В случае если различий между правой и левой частью нет, то число является палиндромом в восьмеричной записи и счетчик палиндромов увеличивается на 1, иначе алгоритм переходит к следующему числу. Так как максимальное значение равно $10^6_{10} = 3\,641\,100_8$, то массив будет содержать максимум 7 элементов.

Теперь, когда алгоритм описан, реализуем его на языке ассемблера для компилятора FASM.

Функции, используемые в реализации алгоритма:

Для вывода результата в консоль будет использована функция `printf`, формирующую данные по заданному шаблону.

Аргументы функции:

- `format` – указатель на C-строку, которая содержит формат результата;
- остальные аргументы – данные, подлежащие форматированию

Для реализации паузы после вывода данных на экран будем использовать функцию `getch`, которая не имеет параметров и возвращает считанный с клавиатуры символ. Она позволит остановить выполнение программы и дать пользователю увидеть ответ

Для завершения работы программы будем использовать функцию `ExitProcess(uint uExitCode)`, которая завершает работу программы.

Она принимает следующие аргументы:

- `uint uExitCode` – определяет код выхода для процесса и для всех потоков, которые завершают работу в результате вызова функции.

Функции реализованные в процессе написания программы

В процессе реализации алгоритма на языке ассемблера программа была разбита на несколько функций:

Главная функция программы. В ней реализован основной цикл, который проходится по всем числам от 1 до 10^6 . Внутри этого цикла вызываются функции `create8Arr(int number)` и `isPalindrom(ref array)` и в случае если число оказывается палиндромом, то увеличивается счетчик, отвечающих за их количество. После цикла на экран выводится значения счетчика.

`void create8Arr(int number):`

Локальные переменные:

- `i` – счетчик
- `copyNum` – переменная, копирующая значение `number`. Создана для того, чтобы не испортить значение `number`
- `pointer` – указатель на текущий элемент массива

Аргументы:

- `number` – число, восьмеричная запись которого будет записана в массив

Заполняет существующий массив `eightsArr` разрядами восьмеричного представления числа `number`. Восьмеричная запись числа находится при помощи цикла с предусловием, который проверяет, что `copyNum > 0`. Внутри же этого цикла в массив записывается значение равное `copyNum mod 8`, а `number` присваивается значение равное `copyNum div 8`.

`int isPalindrom(ref array)`

Локальные переменные:

- `sPtr` – указатель идущий с начала массива
- `ePtr` – указатель идущий с конца массива
- `sElem` – значение ячейки массива на которую указывает `sPtr`
- `eElem` – значение ячейки массива на которую указывает `ePtr`

Аргументы:

- ref array – ссылка на начало массива, хранящего восьмеричную запись числа

Проверяет является ли переданная восьмеричная запись числа в виде массива палиндромом или нет. Если восьмеричное представление является палиндромом, то возвращает в eax 1, иначе 0

Текст программы приведен ниже:

format PE console

include 'win32a.inc'

entry start

```
;Студент: Гуницкий Рон Яковлевич БПИ-195
;Вариант 5
;Условие задачи:
;Разработать программу, определяющую
;число чисел-палиндромов (в восьмеричном
;представлении) в диапазоне от 1 до 10^6
```

section '.data' data readable writable

```
msg1      db 'Count of 8 palindroms: %d',10,0
```

```
arrSize    dd ?           ;размер массива, хранящего 8-е представление чисел
j           dd ?           ;счетчик
count      dd ?           ;количество палиндромов
eightsArr  rd 7           ;ссылка на массив 8-го представления числа
;Константы:
c8          dd 8           ;значение по модулю которого будут искаться палиндромы
maxVal     dd 1000000      ;верхняя граница проверки чисел
```

section '.code' code readable executable

```
;=====MAIN=====
```

start:

```
        mov     [j], 1           ;значение с которого начинается проверка
        mov     [count], 0       ;обнуляем счетчик
```

mainLoop:

```
        mov     ecx, [j]         ;копируем значение счетчика в ecx
        cmp     ecx, [maxVal]    ;сравниваем значение счетчика с maxVal
        jg      endLoop          ;если j >= maxVal
```

```
;создаем массив состоящий из цифр восьмеричного представления
;числа number
```

```
        push    [j]              ;записываем в стек number
        call    create8Arr       ;вызываем функцию create8Arr
        add     esp, 4           ;удаляем переданные аргументы
```

```
;определяем является ли восьмеричное представление палиндромом
        push    eightsArr        ;записываем в стек ссылку на массив
        call    isPalindrom      ;вызываем функцию isPalindrom
        add     esp, 4           ;удаляем переданные аргументы
```

```
;в случае если палиндром - увеличиваем счетчик
        add     [count], eax      ;прибавляем значение eax
        inc     [j]              ;j++
        jmp     mainLoop         ;возвращаемся в начало цикла
```

endLoop:

```
        push    [count]          ;записываем количество в стек
```

```

    push msg1                ;записываем в стек шаблон
    call [printf]            ;выводим сообщение пользователю
    add esp, 8               ;удаляем аргументы

exit:    ;считываем символ и завершаем выполнение программы
    call [getch]
    push 0
    call ExitProcess

;=====

;=====Create8Arr(int number)=====
create8Arr:
;Аргументы функции
number equ ebp+16           ;переданное число

;Локальные переменные
i equ ebp-4                ;счетчик
copyNum equ ebp-8           ;переменная для копии number
pointer equ ebp-12          ;хранит указатель на элемент массива

;сохраняем регистры и выделяем память в стеке под лок. переменные
push eax
push edx
push ebp
mov ebp, esp
sub esp, 12

;инициализируем переменные
mov [i], dword 0           ;обнуляем счетчик
mov edx, [number]          ;копируем значение number
mov [copyNum], edx         ;записываем значение number в copyNum
mov edx, eightArr          ;присваиваем есх ссылку на начало массива
mov [pointer], edx         ;записываем указатель на массив

createArrLoop:
;проверяем, что copyNum не равен нулю
cmp [copyNum], dword 0     ;сравниваем copyNum с нулем
je endCreateArrLoop       ;если равно 0, то выходим из цикла

;делим наше число на константу (8)
mov eax, [copyNum]         ;записываем в eax copyNum (младшие 4 байта)
mov edx, 0                 ;записываем в edx 0 (старшие 4 байта)
div [c8]                   ;делим на 8

;добавляем в массив copyNum % 8
mov [copyNum], eax         ;записываем в copyNum результат деления
mov eax, [pointer]         ;eax = ref array
mov [eax], edx             ;добавляем остаток от деления на 8 в array

;готовим переменные к следующей итерации цикла
inc dword [i]              ;i++
add dword [pointer], 4     ;переход к следующему элементу массива
jmp createArrLoop          ;возвращаемся в начало цикла

endCreateArrLoop:
mov eax, [i]               ;eax = i
mov [arrSize], eax         ;сохраняем размер массива
;возвращаем значения регистров
mov esp, ebp
pop ebp
pop edx
pop eax

ret

```

```
;=====
```

```
;=====IsPalindrom(ref array)=====
```

```
isPalindrom:
```

```
;Аргументы функции
```

```
refArr equ ebp+16 ;ссылка на начало массива
```

```
;Локальные переменные
```

```
sPtr equ ebp-4 ;указатель идущий с начала
```

```
ePtr equ ebp-8 ;указатель идущий с конца
```

```
sElem equ ebp-12 ;значение элементов с начала
```

```
eElem equ ebp-16 ;значение элементов с конца
```

```
;сохраняем регистры и выделяем память в стеке под лок. переменные
```

```
push ecx
```

```
push edx
```

```
push ebp
```

```
mov ebp, esp
```

```
sub esp, 16
```

```
;инициализируем переменные
```

```
mov edx, [refArr] ;присваиваем ecx ссылку на начало массива
```

```
mov [sPtr], edx ;записываем ссылку на массив
```

```
mov edx, [edx] ;получаем значение элемента
```

```
mov [sElem], edx ;записываем первый элемент массива
```

```
mov ecx, [arrSize] ;получаем ссылку на конец+4 массива
```

```
dec ecx ;вычитаем 1 из размера массива
```

```
imul ecx, 4 ;получаем количество байт, на которое надо перейти
```

```
mov edx, [sPtr] ;копируем ссылку на первый элемент
```

```
add edx, ecx ;получаем ссылку на конец массива
```

```
mov [ePtr], edx ;записываем ссылку на конец массива
```

```
mov edx, [edx] ;получаем значение элемента
```

```
mov [eElem], edx ;записываем последний элемент массива
```

```
isPalindromLoop:
```

```
;проверяем отношения ссылок на элементы
```

```
mov edx, [sPtr] ;записываем ссылку на элемент массива
```

```
cmp edx, [ePtr] ;сравниваем ссылку с конца и ссылку с начала
```

```
jge itIsPalindrom ;если sPtr >= ePtr то это палиндром
```

```
;проверяем равенство символов
```

```
mov edx, [sElem] ;записываем значение элемента с начала
```

```
cmp edx, [eElem] ;сравниваем sElem с eElem
```

```
jne itIsNotPalindrom ;если sElem != eElem, то это не палиндром
```

```
add dword [sPtr], 4 ;сдвигаем указатель на следующий элемент
```

```
sub dword [ePtr], 4 ;сдвигаем указатель на элемент назад
```

```
mov edx, [sPtr] ;edx = sPtr
```

```
mov edx, [edx] ;получаем значение элемента
```

```
mov [sElem], edx ;записываем значение массива
```

```
mov edx, [ePtr] ;edx = ePtr
```

```
mov edx, [edx] ;получаем значение массива
```

```
mov [eElem], edx ;записываем значение массива
```

```
jmp isPalindromLoop ;возвращаемся в начало цикла
```

```
itIsPalindrom:
```

```
mov eax, 1 ;возвращаем 1
```

```
jmp endIsPalindrom ;идем в конец функции
```

```
itIsNotPalindrom:
```

```
mov eax, 0 ;возвращаем 0
```

```
endIsPalindrom:
```

```
;возвращаем значения регистров
```

```
mov esp, ebp
```

```
        pop    ebp
        pop    edx
        pop    ecx
ret
;=====

section '.idata' data readable import

        library kernel, 'kernel32.dll',\
                msvcrt, 'msvcrt.dll'

        import kernel,\
                ExitProcess, 'ExitProcess'

        import msvcrt,\
                printf, 'printf',\
                scanf, 'scanf',\
                getch, '_getch'
```

Тестирование

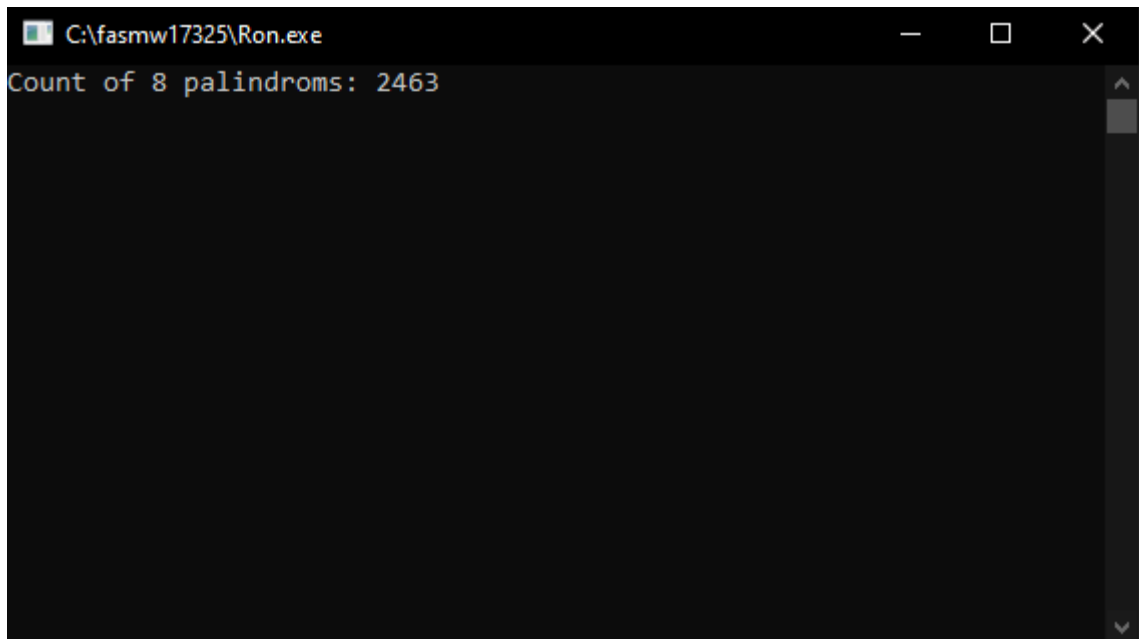


Рисунок 1 – Тестирование программы

Программа работает корректно и всегда выводит правильный результат (см. рис. 1)

Список используемых источников

1. SoftCraft «Программирование на языке ассемблера. Микропроект. Требования к оформлению. 2020-2021 уч.г.» (<http://softcraft.ru/edu/comparch/tasks/mp01/>)
2. natalia.appmat «Программирование на языке ассемблера» (<http://natalia.appmat.ru/c&c++/assembler.html>)
3. osinavi « Команды передачи управления» (<http://osinavi.ru/asm/4.html>)
4. vsokovikov.narod «Функция ExitProcess» ([http://vsokovikov.narod.ru/New MSDN API/Process thread/fn_exitprocess.htm](http://vsokovikov.narod.ru/New_MSDN_API/Process_thread/fn_exitprocess.htm))
5. Википедия (2020) «Палиндром» (<https://ru.wikipedia.org/wiki/%D0%9F%D0%B0%D0%BB%D0%B8%D0%BD%D0%B4%D1%80%D0%BE%D0%BC>)
6. Flat assembler « Macro to perform a modulo with div instruction» (<https://board.flatassembler.net/topic.php?t=20704>)