

title	subtitle	author	lang	toc-title	bibliography	csl	toc	toc-depth	lof	
Отчёт по лабораторной работе №8	Простейший вариант	Волчкова Елизавета Дмитриевна	ru-RU	Содержание	bib/cite.bib	pandoc/csl/gost-r-7-0-5-2008-numeric.csl	true	2	true	1

Цель работы

Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

Задание

Напишите программу, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$, т.е. программа должна выводить значение $f(x_1) + f(x_2) + \dots + f(x_n)$. Значения x_i передаются как аргументы. Вид функции $f(x)$ выбрать из таблицы 8.1 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу на нескольких наборах $x = x_1, x_2, \dots, x_n$.

Выполнение лабораторной работы

Описываются проведённые действия, в качестве иллюстрации даётся ссылка на иллюстрац

выводит значение регистра esx. Внимательно изучите текст программы (листинг 8.1).

root@liza2006-VirtualBox: ~/work/arch-pc/lab08

```

lab08
root@liza2006-VirtualBox:~/work/arch-pc# touch lab8-1.asm
root@liza2006-VirtualBox:~/work/arch-pc# cd lab08
root@liza2006-VirtualBox:~/work/arch-pc/lab08# ls
root@liza2006-VirtualBox:~/work/arch-pc/lab08# touch lab8-1.asm
root@liza2006-VirtualBox:~/work/arch-pc/lab08# ls
lab8-1.asm
root@liza2006-VirtualBox:~/work/arch-pc/lab08#

```

ию

текст программы из листинга 8.1. Затем создала исполняемый файл и проверила его работу.

Данный пример показывает, что использование регистра esx в теле цикла loop может привести к некорректной работе программы. Далее я изменила текст программы добавив изменение значение регистра esx в цикле.

Потом создала исполняемый файл и проверила его работу.

```
//root/work/arch-pc/lab08/lab8-1.asm 637/637 100%
#include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
; ---- Вывод сообщения 'Введите N: '
mov eax,msg1
call sprint
; ---- Ввод 'N'
mov ecx, N
mov edx, 10
call sread
; ---- Преобразование 'N' из символа в число
mov eax,N
call atoi
mov [N],eax
; ----- Организация цикла
mov ecx,[N] ; Счетчик цикла, 'ecx=N'

label:
mov [N],ecx
mov eax,[N]
call iprintLF ; Вывод значения 'N'
loop label ; 'ecx=ecx-1' и если 'ecx' не '0'
; переход на 'label'
call quit

1Help 2UnWrap 3Quit 4Hex 5Goto 6 7Search 8Raw 9Format 10Quit
```

Какие значения принимает регистр ecx в цикле? Соответствует ли число проходов цикла значению N введенному с клавиатуры?

После внесла изменения в текст программы, добавив команды push и pop (добавления в стек и извлечения из стека) для сохранения значения счетчика цикла loop. Создала исполняемый файл и проверила его работу. Соответствует ли в данном случае число проходов цикла значению N введенному с клавиатуры? - нет! Циклов было больше 100.

Далее создала файл lab8-2.asm в каталоге ~/work/arch-pc/lab08 и ввела в него текст программы из листинга 8.2. Потом создала исполняемый файл и запустила его, указав аргументы: Сколько аргументов было обработано программой? -

Затем создала файл lab8-3.asm в каталоге ~/work/arch-pc/lab08, ввела в него текст программы из листинга 8.3.

Я создала исполняемый файл и запустила его, указав аргументы.

Написала программу, которая находит сумму значений функции $f(x)$ для $x = x_1, x_2, \dots, x_n$, т.е. программа должна выводить значение $f(x_1) + f(x_2) + \dots + f(x_n)$. Значения x_i передаются как аргументы. Вид функции $f(x)$ выбрала из таблицы 8.1 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7 и создала исполняемый файл и проверила его работу на нескольких наборах $x = x_1, x_2, \dots, x_n$

```
/root/work/arch-pc/lab08/lab8-dz.asm [----] 0 L: 63+26 89/ 94] *(3242/3410b) 0010 0x00A [*](X)

.convert_to_string:
    dec edi                ; Move backwards in the buffer
    xor edx, edx           ; Clear edx (used for division)
    mov ebx, 10            ; Divisor
    div ebx                ; Divide eax by 10
    add dl, '0'            ; Convert remainder to ASCII
    mov [edi], dl          ; Store character in the buffer
    test eax, eax          ; Check if quotient is zero
    jnz .convert_to_string ; If not zero, continue converting

    ; Calculate the length of the string
    mov edx, sum_res       ; Start of the result string
    sub edx, edi           ; Length of the string

    ; Print the result
    mov ecx, edi           ; Pointer to the start of the result string
    mov ebx, 1             ; File descriptor (stdout)
    mov eax, 4             ; Syscall number for sys_write
    int 0x80               ; Call kernel

    ; Print a new line
    mov edx, 1             ; Length of the new line
    mov ecx, new_line      ; Pointer to new line
    mov eax, 4             ; Syscall number for sys_write
    int 0x80               ; Call kernel

    ; Exit the program
    mov eax, 1             ; Syscall number for exit
    xor ebx, ebx           ; Return 0
    int 0x80               ; Call kernel

1Help 2Save 3Mark 4Replac 5Copy 6Move 7Search 8Delete 9PullDn 10Quit
```

Выводы

Целью работы было приобретение навыков написания программ с использованием циклов и обработкой

аргументов командной строки, проделав данные задания я усвоила материал, имею представления о том, как правильно использовать циклы и командные строки.

Список литературы{.unnumbered}

1. GDB: The GNU Project Debugger. — URL: <https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 c. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 c. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 c. — ISBN 9781784396879.
9. Колдаев В. Д., Lupin С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
13. Робачевский А., Немнюгин С.,

Стесик О. Операционная система UNIX. — 2-е изд. — БХВ- Петербург, 2010. — 656 с. — ISBN 978-5-94157-538-1. 14. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix. 15. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science). 16. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science). ∴ {#refs} ∴