Practical1

```
section .data
   msg_prompt db 10,'Enter 85 bytes of input: ',10   ; Prompt message
for input
   msg_output db 10, 'Data received and output: ',10   ; Message before
output
section .bss
   array resb 200          ; Reserve 200 bytes for input data
   counter resb 1          ; Counter for iterations
section .text
global _start
_start:
   ; Print the prompt message to the user
   mov rax, 1              ; Syscall number for write (stdout)
   mov rdi, 1              ; File descriptor 1 (stdout)
   mov rsi, msg_prompt      ; Address of the prompt message
   mov rdx, 25             ; Length of the prompt message
   syscall
   ; Initialize the counter to 5 for the loop
   mov byte [counter], 5
   mov rbp, array          ; Load address of 'array' into RBP
   ; Loop 1: Read 17 bytes at a time into 'array' from standard input
loop1:
   mov rax, 0              ; Syscall number for read (stdin)
   mov rdi, 0              ; File descriptor 0 (stdin)
   mov rsi, rbp            ; Buffer to store input data (address of
'array')
   mov rdx, 17             ; Number of bytes to read
   syscall

   add rbp, 17             ; Move buffer pointer forward by 17 bytes
   dec byte [counter]       ; Decrease the counter
   jnz loop1               ; If counter is not zero, repeat the loop
   ; Print the output message to the user
   mov rax, 1              ; Syscall number for write (stdout)
   mov rdi, 1              ; File descriptor 1 (stdout)
   mov rsi, msg_output      ; Address of the output message
   mov rdx, 24             ; Length of the output message
   syscall
   ; Initialize the counter again to 5 for the second loop
   mov byte [counter], 5
   mov rbp, array          ; Load address of 'array' into RBP
   ; Loop 2: Write 17 bytes at a time from 'array' to standard output
loop2:
   mov rax, 1              ; Syscall number for write (stdout)
   mov rdi, 1              ; File descriptor 1 (stdout)
   mov rsi, rbp            ; Buffer with data to write (address of
'array')
   mov rdx, 17             ; Number of bytes to write
   syscall

   add rbp, 17             ; Move buffer pointer forward by 17 bytes
   dec byte [counter]       ; Decrease the counter
   jnz loop2               ; If counter is not zero, repeat the loop
; Exit the program
mov rax, 60
```

xor rdi, rdi
syscall
outpout
; Syscall number for exit
; Exit status 0

Output

```
rllab@fedora:/home/liveuser$ nasm -f elf64 prathamesh1.nasm
rllab@fedora:/home/liveuser$ ld -o prathamesh1 prathamesh1.o
rllab@fedora:/home/liveuser$ ./prathamesh1

Enter 85 bytes of input:333333
555555
554444
999999
444444

Data received and outpu333333
555555
554444
999999
444444
rllab@fedora:/home/liveuser$ ./prathamesh1

Enter 85 bytes of input:7454544
A4B3C6C7D2F
ABCDE454F54
58659989555
6559

Data received and outpu7454544
A4B3C6C7D2F
ABCDE454F54
58659989555
6559
rllab@fedora:/home/liveuser$ ./prathamesh1

Enter 85 bytes of input:ACDFBA45
55543434
34345441
89767559
98844465

Data received and outpuACDFBA45
55543434
34345441
89767559
98844465
rllab@fedora:/home/liveuser$
```