NAME: AMMAAR AHMAD

ROLL NO: - 1801CS08

PURPOSE OF THE LAB:

TO FAMILIARIZE WITH NASM AND x86 ASSEMBLY LANGUAGE AND IT’S DIFFERENT INSTRUCTIONS.

ABOUT THE GAME

IT’S A SIMPLE NUMBER GAME WHERE IN EACH TURN EITHER COMPUTER OR USER CAN WITHDRAW ANY NUMBER OF BALLS PROVIDED NUMBER = 2^X FOR SOME X BELONGS TO WHOLE NUMBER AND LOG(NUMBER)>=X.

USER WILL ENTER THE NUMBER OF BALLS TO START THE GAME

COMPUTER WILL HAVE THE FIRST TURN

SOURCE CODE:

; Simple Game using Nasm

; CS/Algo used: Simple Application of Number Theory

%include "io.mac"

.DATA

Start\_msg db "Please follow the rules of the games provided below",0

Rule\_1 db "In each step only 2^x balls can be taken away where 0<=x<=log(No. of balls left)",0

Rule\_2 db "Player who cannot take the ball at any step has lost the game",0

Player\_Won db "You Won",10 ,0

Computer\_Won db "Computer Won",10 ,0

Restart\_msg db "Do you want to play again: Y for Yes, N for No: ",0

Game\_msg db "Enter the number of balls to start with: ",0

Playgame\_msg db "Press any key to continue: ",0

Ingame\_msg db "No of balls left: ",0

Continue\_msg db "Pick the balls: ",0

Error\_msg db "Wrong Input, Please try Again: ",10 ,0

.CODE

.STARTUP

;Instructions

PutStr Start\_msg ;Printing Message for user

nwln ;Printing New Line

;Printing Rules

PutStr Rule\_1

nwln

PutStr Rule\_2

nwln

;Main Game

PutStr Playgame\_msg ;Press any key to enter

GetCh AL ;Key to start the game

;Start

NEW: PutStr Game\_msg ;Number of balls from user

GetInt AX ;Number of balls

MOV BX, 03 ;BX=3

GAME: ;Checking if remaining number of balls is 2^x

PUSH AX ;Store AX value in stack

MOV DX, AX ;DX=AX

DEC DX ;Decrementing DX

AND DX, AX ;Checking whether AX is 2^x

JZ LOST ;IF ZF=1 Player Lost

;If not 2^x continue game

POP AX ;AX = top of stack

PUSH AX ;Store AX value in stack

XOR EDX, EDX ;Initialing EDX=0

AND EAX, 00FFH ;Initialing EAX = AX

DIV BX ;Divide BX

MOV CX, DX ;Copying Remainder to ECX

;Remaining Number of balls in AX

POP AX ;AX = top of stack

;Computer Move

CMP CX, 00 ;Check if CX=0

JNZ NEXT ;If ZF=1 then jump to next

DEC AX ;Else decrement AX

JMP STEP ;Unconditional jump to Step

NEXT: SUB AX, CX ; AX = AX - CX

JZ LOST ; IF AX=0 jump to LOST label

;Present Game Situation

STEP: PutStr Ingame\_msg ;Printing Current number of balls

PutInt AX

nwln ;NewLine

PutStr Continue\_msg ;Asking user for their turn

GetInt DX ;User choice

CMP AX,DX ;Comparing AX with DX

JZ WON ;If equal then User Won

JNC SKIP ;If DX<AX Jump to Skip

;Printint Error Message if DX>AX

PutStr Error\_msg

JMP STEP

;Checking if DX is of 2^x form

SKIP: MOV CX, DX ;Copying DX to CX

DEC CX ;Decrement CX

AND CX, DX ;num & (num-1)

JZ AHEAD ;if ZF=1 it is of 2^x form, jump to Ahead

;If not 2^x print error message

PutStr Error\_msg ;Error message

JMP STEP ;If error go back for user choice

;Game Update after User Choice

AHEAD: SUB AX, DX ;AX=AX-DX

JMP GAME ;Game continues

;Player won the Game

WON: PutStr Player\_Won

nwln

JMP RESTART ;Jump to restart

;Computer won the Game

LOST: PutStr Computer\_Won

nwln

;Restart Game

RESTART:PutStr Restart\_msg

GetCh AL ;User choice input

CMP AL, 89 ;Checking if AL ='Y'

JE NEW ;If equal restart->NEW

nwln

;Exit Game

EXIT MOV EBX, 0 ; return 0 status on exit - 'No Errors'

MOV EAX, 1 ; invoke SYS\_EXIT (kernel opcode 1)

INT 80H

