RELATION OF PROJECT: COMPUTER ARCHITECTURE

PROJECT : Write two old exams in assembler 8086 (versions 16 bits emu 8086 (using Masm to compile them) and versions 64 bits (using Nasm to compile them))

if you want the exams test and complete code look at the link :
https://github.com/SandroSartoni/CAs-Exam

NASM (64 BITS) VERSION :

1-How to install NASM ON UBUNTU

You can do it following the video: https://www.youtube.com/watch?v=ooJfdrds9Dk

during the installation you must move from User to Super User : you can follow the instruction here : https://www.youtube.com/watch?v=JmLucEh8U_8
after complete installation of Nasm,

2- Use any test editor (gedit , vim ..) to write your code but your file must have extention .asm

3- How to Compile and Run your Code

you can use this video at the following link:
https://www.youtube.com/watch?v=AujBSIeIGmE&feature=youtu.be

or follow these steps

*** nasm -f elf64FILE_NAME.asm

*** ld -s -o FILE_NAME FILE_NAME.o

*** ./ FILE_NAME

elf64 means: you are compiling your code in 64bits (you can put 32 if you want to do it on 32 bits)

BUS: SPEUDO CODE

DECLARE_DATA()

CALL READ_DEPARTURE_TIME()

PREPARE_PARAMETER_FOR_FIRST_PATH ()

CALL CALCULATE_TAVEL()

PREPARE_PARAMETER_FOR_SECOND_PATH ()

CALL CALCULATE_TAVEL()

END

PROC CALCULATE_TRAVEL()

CALL FIND_AVAILABLE_TIME_IN_VECT_1()

ADD_TIME_FOR_THE_TRAVEL

CALL FIND_AVAILABLE_TIME_IN_VECT_2()

ADD_TIME_FOR_THE_TRAVEL

CALCULATE THE DIFFERENCE BETWEN_DEPARTURE_HOUR_AND_ARRIVAL_HOUR

END_PROC

There are some procedures like Print_new_line , Print_Line...

FOOTBALL : SPEUDO CODE

DECLARE_DATA()

PREPARE_PARAMETER_FOR_FIRST_GROUP

CALL SCAN() : allows to propose all possible match in a group and get their scoore and assign to each Team its number of points according to the rules of the competition.

CALL SORT() : order each group according to the number of point and name

CALL PRINT() : print the classification of each group

repeat that for the seond and third group

FIND_THE_FOURTH_FOR_THE_SEMI_FINAL , among the seconds of each group find the best to have 4 teams and make the semi_final and after the final

PREPARE_PARAMETER_FOR_1ST_SEMI_FINAL

CALL SEMI_FINAL () : take in input 2 teams and return the winner

PREPARE_PARAMETER_FOR_2ST_SEMI_FINAL

CALL SEMI_FINAL () : take in input 2 teams and return the winner

PREPARE_PARAMETER_FOR_THE_FINAL

CALL SEMI_FINAL () : take in input 2 teams and return the winner

PRINT THE WIENNER
END

Sartoni Sandro && Wamba Briand

EXEMPLE OF NASM CODE

```
section .data   ;In this section I declare data that needs to be iniziliased

var db 20

section .bss     ; in this session I declare data without inizializing them

var resb 8       ; ( means reseve 8 bytes) res( reserve)  b(byte) it can be d (double),w...

section .text

   global _start:              ;used to tell the starting point of my program

_start:

  push rax           ; I save into the stack the value of rax
  mov rax,1          ; I put into rax the value 1
  mov rbx,1          ;  ----
  add rax,rbx        ; I compute additon betwen rax and rbx and store the risult in rax -> rax=rax+rbx
  ---
  ---
  pop rax            ; I retrieve my value save previously, taking it from the stack by operation pop


   mov rax,1  these 3 Line are directives to end your code correctly
   mov rbx,0
   int 80h
```

In EMU8086 I encontered this ERROR THE EMULATOR IS HALTED and on Internet I didn’t find the solution
but to solve it just ADDED the directive RET at the end of the procedures