Title : Project 3
Point Value : 100
Deadline : 11/12 at 11.59pm

Topics Covered:

Introduction to NASM, x86-64 instructions and architecture, Loops, System calls, ASCII

Project Description:

For this project we will be implementing basic string manipulation in x86-64 assembly.

Write and submit a x86-64 assembly language program “reverse.asm” that:

- Asks the user for a string
  - The string from the user should be more than 8 characters long (spaces included)
  - You have to do error checking that the string is longer than 8 characters
- Asks the user for a number between 2 and the maximum number of characters from the string they entered.
  - You have to do error correction
  - This will be the location to split the string in the reverse section

- Displays the unedited string

- And finally displays an edited string to the user edited in the following way:
  - Divide the input string into two sections starting at the location entered by the user
  - Then reverse each half
  - Finally put these halves back into one string.
  - For instance, if the input text is “An apple a day keeps the doctor away!” and the entered location is 18 then after applying reverse to it the result would be: “*eek yad a elppa nA!yawa rotcod eht sp”*.
  - The newline should be kept at the end of the string and is not considered part of the string.

Below are sample runs of the program:

Sample run 1:

```
Please enter the text:
An apple a day keeps the doctor away!
Enter a number between 2 and the total number of characters in the string
18
```

```
Unedited String: An apple a day keeps the doctor away!
Edited String: eek yad a elppa nA!yawa rotcod eht sp
```

Sample run 2:

```
Please enter the text:
An apple a day keeps the doctor away!
Enter a number between 2 and the total number of characters in the string
1
Enter a number between 2 and the total number of characters in the string
45
Enter a number between 2 and the total number of characters in the string
4
Unedited String: An apple a day keeps the doctor away!
Edited String: a nA!yawa rotcod eht speek yad a elpp
```

Notes:

- Remember characters/numbers are read in ASCII.
    - You have to tackle double digit numbers.

General Project (Hard) Requirements:

- **Code that does not assemble will receive 50% off**
- **We are programming for 64 bit Intel Architecture**
    - **Code written for 32 bit Intel will receive 20 points off**
- **You can work individually or with another person for this project.**
    - **No more than 2 people can work together.**
- You must have a comment at the top of the code detailing what the code does.
- This comment should also include your full name and user ID.
    - If working with 2 people both names and user IDs should be included.
- You must use good coding style with respect to variable names, spacing, labels and comments.
- Submit **only the .asm file named reverse.asm** to blackboard.
    - Late submissions will accrue a deduction of 10 points for every 6 hours it is late (Outside of the late days).
- You cannot use C/C++ function calls. **You have to use system calls**
- Any late submissions will incur a penalty of 10 points per every 6 hours they are late.(Outside of the late days)

Grading Breakdown: (remember that code that doesn't compile will not get full credit)

- [75] Functionality
    - [20]Display prompts to user and read in values
    - [10]Display unedited string
    - [10]Display edited string

  - [20]Text edit functionality
  - [5]Use of system calls
  - [5]Program exits correctly (No Seg Faults)
  - [5]No extra new lines or spaces printed
- [25] Style (only if it compiles)
  - [5]Comments where necessary
  - [5]Code written for 64 bit Intel Assembly
  - [5]Comment at top of program with name, user id and explanation of code
  - [5]Code easy to read
    - Good variable names
    - Indentation
  - [5]Submitted correctly