CS2.201: Computer Systems Organization
Professor: Praveen Paruchuri
Spring 2023
International Institute of Information Technology, Hyderabad

Assignment 1

Deadline: 23:55, 18/05/2023

Welcome to Assignment 1 of the Computer Systems Organisation Course. The aim of this assignment is to familiarize you with writing x86 code. On completion of this assignment you should be able to successfully write arithmetic, conditional, looping components, and conditional jumps in x86-64.

Instructions: Read all the instructions below carefully before you start working on the assignment.

- Writing complete code with successful execution guarantees full marks. Failure on test cases will result in penalisation. Therefore ensure all edge cases are handled.
- Strict plagiarism checks will be performed on all submissions. Any and all forms of plagiarism will result in zero marks for this assignment.
- Write well-organised code using procedures for repeated operations.
- Hard coded solutions will get a straight zero.
- Comment every line of your code and justify why you write that statement. Total of 5 marks for commented code.
- Total marks for assignment is 50(30 marks for working code submission, 20 marks for viva).
- C files are given for each question. Write the assembly code for the required function declared in these files. **DO NOT** modify the given C files.

Submission format: Strictly adhere to the following submission format. Failure to do so may result in an erroneous evaluation of your assignment.

- The following directory structure is expected,

```
./<roll_number>
├── q1
│   ├── q1.s
│   └── q1.c
├── q2
│   ├── q2.s
│   └── q2.c
├── q3
│   ├── q3.s
│   └── q3.c
├── q4
│   ├── q4.s
│   └── q4.c
└── q5
    ├── q5.s
    └── q5.c
```

- Zip the ./<*roll_number*> folder and name the zipped folder as <*roll_number*>_assign1.zip

Assume all the integer variables to be long long int.

Problem 1: 5 marks

Given an array of non-negative integers, find the sum of all numbers which are divisible by 3 in it. You are allowed to use any x86 instruction taught in class for this question.

Input/Output Format

- INPUT: Contains two lines. First line contains a single value N, size of the array. $(1 \leq N \leq 50)$ The second line contains N values $a_1, a_2 \dots a_n$, elements of the array. $(0 \leq a_i < 10^9)$
- OUTPUT: M, where M is the sum of numbers which are divisible by 3

Sample Test Case

Input:
5
1 12 5 7 3
Output:
15

Problem 2: 5 marks

Given four values a, x, b, y calculate $(a.2^x + b.2^y)$. Do not use inbuilt *imul* , *leaq* or similar instructions for this question.

Input/Output Format

- INPUT: a x b y $(-10^4 \leq a, b \leq 10^4\ ;\ -10 \leq x, y \leq 10)$
- OUTPUT: V, where V is calculated as mentioned above.

Sample Test Case

Input: 5 3 3 -1
Output: 41

Problem 3: 6 marks

Given two numbers M and N compute the quotient and remainder for M/N. Do not use inbuilt *idiv* or similar operations. If the denominator is zero, return -1 -1.

Input/Output Format

- INPUT: M N $(-10^4 \leq M, N \leq 10^4)$
- OUTPUT: **q, r**; where q is the quotient and r is the remainder. $(0 \leq r < N;\ N * q + r = M)$

Sample Test Cases

Input: 153 5
Output: 30 3

Input: -31 5
Output: -7 4

Problem 4: 7 marks

Given a linked-list of a fixed size, calculate and return the product of all the values stored in the linked-list.
The C file given for this question contains code for the initialization of the linked-list. This file also contains the declaration of a function named *get_product_assembly*(). Write assembly code for this function, which performs the task mentioned above. Carefully go through the C file to obtain hints on how to approach this question. You are allowed to use any x86 instruction taught in class for this question.

Sample Test Case
$Enter\ 10\ values: -2\ 2\ 2\ 2\ 2\ 2\ 2\ 2\ 2\ 2$
$Product\ of\ all\ the\ values\ in\ the\ linked\ list: -1024$
$Product\ of\ all\ the\ values\ in\ the\ linked\ list\ (assembly\ code): -1024$

Bonus : Handle the overflow that might occur when the linked-list contains large values. So, in the case where overflow occurs, you can return (product % LLONG_MAX). (2 marks)

Problem 5: 7 marks

Given an array of numbers, your function should return the address of the smallest number in the array. The given C file de-references the address returned by your function and prints the value stored at that address. You are allowed to use any x86 instruction taught in class for this question.

- INPUT: Contains two lines. First line contains a single value N, size of the array. $(1 \leq N \leq 50)$ The second line contains N values $a_1, a_2 \ldots a_n$, elements of the array. $(-10^9 < a_i < 10^9)$
- OUTPUT: **S**, the smallest value in the array

Sample Test Case
Input :
3
2 99 -3
Output: -3

All the Best!