Department of Computer Science

COS284 Practical and Assignment 2: System Calls and Basic Math

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

1 Introduction

This document contains both practical 2 and assignment 2. In general the assignments will build upon the work of the current practical.

1.1 Submission

The Assignment will be fitchfork only and is due at 17:00 on Friday 4 September. You may use the Discord server to ask for assistance with the assignment and practical components.

1.2 Additional Resources

1.2.1 Ascii Table

You can find an Ascii table here for your use with the practical and assignment tasks: `http://www.asciitable.com/`.

1.2.2 GDB

GDB (GNU Debugger) is a debugging tool that allows you to pause program execution and directly view the content of memory and registers. As the inner functioning of assembly can be hard to tell from program behaviour or output, GDB is a useful tool to aid you in understanding exactly what your assembly code is doing at any specific point. The following links contain informative content pertaining to the operation of GDB:

- Here is a tutorial to install GDB on your Linux system: `https://www.tutorialspoint.com/gnu_debugger/installing_gdb.htm`.
- A cheat sheet detailing some of the basic commands of GDB: `http://web.cecs.pdx.edu/~apt/cs491/gdb.pdf`. The print registers command is particularly useful in debugging assembly
- A video tutorial on using GDB to debug a simple program: `https://www.youtube.com/watch?v=bWH-nL7v5F4`. While the program being debugged is in C the general principle is the same. You invoke the gdb command with the name of your executable and use the exact same commands to debug your assembly code. It should be noted that you need not include any additional arguments in your compile command to allow the use of GDB with your executable, unlike with C. Another note is that when setting breakpoints in assembly you should use the line numbers to set the breakpoints, not labels.

1.3 Plagiarism policy

It is in your own interest that you, at all times, act responsible and ethically. As with any work done for the purpose of your university degree, remember that the University of Pretoria will not tolerate plagiarism. Do not copy a friend's work or allow a friend to copy yours. Doing so constitutes plagiarism, and apart from not gaining the experience intended, you may face disciplinary action as a result.

For more on the University of Pretoria's plagiarism policy, you may visit the following webpage: `http://www.library.up.ac.za/plagiarism/index.htm`

1.4 Practical component [28%]

1.4.1 Task 1 [8%]

For the first task of the practical you will be required to implement a simple cat program. A cat program simply takes it's input and returns it as output. In this cat program you must prompt the user to enter a 5 digit number, and must then simply output the 5 digit number, unaltered. An example output would be:

```
Please input a 5 digit number: 55123
This is the number you are looking for: 55123
```

You can assume that all inputs will be 5 digits long. When you have completed the task you must create a tarball containing your source code file named task1.asm and upload it to the Practical: Task 1 upload slot on the CS website.

1.4.2 Task 2 [10%]

In this task you will be expected to produce code that will prompt the user to input a single digit number, subtract a constant of 5 from the number and determine whether the result is a positive or negative integer, outputting an appropriate message afterwards. Note: in order to work with the actual numerical value of integer input, you will have to convert from the Ascii value to the real integer value. E.g:

```
Please input an integer: 6
Result: positive

Please input an integer: 3
Result: negative
```

Note: for the purposes of this assignment we will consider a result of 0 to be positive.You can assume that all inputs will be 1 digit long. When you have completed the task you must create a tarball containing your source code file named task2.asm and upload it to the Practical: Task 2 upload slot on the CS website.

1.4.3 Task 3 [10%]

In this task you must create a program that will prompt for and then take two single digit numbers as input, add them together and display the output. For the purposes of this task you may assume that all sums to be output will only be a single digit (less than 10). Note, similarly to the previous task, you must convert your output back to ascii from its integer value. E.g:

```
Enter the first number: 3
Enter the second number: 2
5
```

You can assume that all inputs will be 1 digit long. When you have completed the task you must create a tarball containing your source code file named task3.asm and upload it to the Practical: Task 3 upload slot on the CS website.

1.5 Assignment component [72%]

1.5.1 Task 1 [24%]

For task 1 of the assignment you must implement a program that extends the functionality of the final practical task by adding any two single digits supplied by the user.

Input: Your program must ask the user for a single digit number with the prompt "Please enter the first number: ". You must then read the character given by the user. You can assume it will always be a single digit. You must prompt the user for another number with the message "Please enter the second number: ". Again you may assume the input will be a single digit.

Processing: Given the two digits you must convert each to their numerical value from their ascii value (see `asciitable.com` for ascii character value mappings) and add the two numbers.

Output: You must output the number produced by adding the two user input values. Your output should always be 2 characters. If the addition results in a single digit answer then pad the output with a '0'.

Examples:

```
Please enter the first number: 2
Please enter the second number: 4
06

Please enter the first number: 9
Please enter the second number: 6
15
```

When you are finished, create a tarball containing your source code file named **main.asm** and upload it to the assignments.cs.up.ac.za website, under the **Assignment 2 Task 1** submission slot.

1.5.2 Task 2 [48%]

Task 2 of the assignment component requires you to implement an asm program in file named 'main.asm'. Your program will be required to ask the user to input 2 double digit numbers and output the quotient and remainder resulting from the division of the numbers.

Input: Your program must ask the user for input by printing out the message "Please enter the first number: ". You must then read the input given by the user. You can assume that the user will always input exactly 2 characters which are digits '0'-'9'. After reading in the first number you must ask for the second number by printing "Please enter the second number: ". You can also assume the second input will only be 2 characters but with an additional restriction that the second input will not be '00'.

Processing: The input you will be given will be in ascii format which means you must convert the ascii characters into a numerical value. You can refer to `asciitable.com` for the mapping between ascii characters and their numerical value. Once you have both numbers converted from ascii strings to numerical values you must divide the first number by the second.

Output: Finally you must output the quotient and the remainder of dividing the first number by the second. Your output should always be exactly 5 characters. 2 characters should be used for the quotient and 2 should be used for the remainder. If either the quotient or remainder are single digit numbers you should pad the output with a zero. You should also output an 'r' between the quotient and remainder numbers.

Examples:

```
Please enter the first number: 77
Please enter the second number: 02
38r01

Please enter the first number: 03
Please enter the second number: 55
00r03

Please enter the first number: 33
Please enter the second number: 11
03r00

Please enter the first number: 32
Please enter the second number: 11
02r10
```

When you are finished, create a tarball containing your source code file named **main.asm** and upload it to the assignments.cs.up.ac.za website, under the **Assignment 2 Task 2** submission slot.

2 Mark Distribution

Activity	**Mark**
Practical	28
Assignment	72
Total	**100**