Contents

1 Introduction 1
  1.1 Submission .................................................. 1
  1.2 Plagiarism policy ............................................ 1
  1.3 Practical component [25%] ................................ 2
    1.3.1 Task 1: Looping - 5 marks [5 %] ...................... 2
    1.3.2 Task 2: ASCII converter - 20 marks [20 %] ........... 2
  1.4 Assignment component - 40 Marks [75%] .................. 3
    1.4.1 Assignment Notes ....................................... 3

2 Mark Distribution 4

1 Introduction

This document contains the both practical 3 and assignment 3. In general the assignments will build upon the work of the current practical.

1.1 Submission

The practical component must be uploaded to fitchfork during or before your practical session.

You have until Friday the 11th of September at 17:00 to complete the assignment component. **No late submissions will be accepted.** The practical upload slots will become available on the Monday the 31st of August at 11:00.

1.2 Plagiarism policy

It is in your own interest that you, at all times, act responsibly 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](http://www.library.up.ac.za/plagiarism/index.htm)
1.3 Practical component [25%]

1.3.1 Task 1: Looping - 5 marks [5 %]

For this task you must implement an assembly program that does the following:

1. Accepts one single digit input from the standard input stream
2. Displays the 10 consecutive numbers starting from the input
3. The numbers must be in base 36 (0-9,a-z)
4. once ‘z’ is reached the program should stop counting.
5. Employ a looping structure to realize the points above

Remember to use _start as your program’s entry point.

Example execution of your program is shown:

alex@linux-PC:/home/Documents$ ./count.out
0
alex@linux-PC:/home/Documents$ ./count.out
f
alex@linux-PC:/home/Documents$ ./count.out
w
alex@linux-PC:/home/Documents$/

When you have finished, create a tarball or zip containing only your source code and upload it to Fitchfork on the CS website, using the Practical 3 Task 1 upload link. You have 5 uploads for this task. Name your assembler file count.asm.

1.3.2 Task 2: ASCII converter - 20 marks [20 %]

For this task you must implement an assembly program that does the following:

1. Accepts a two hex digit input (00-ff)
2. Displays ASCII character that corresponds with input hex value.
3. Applies bitwise NOT operation to the input.
4. Divides the result of the previous step by 2.
5. Displays the ASCII character that corresponds to this new value.

Remember to use _start as your program’s entry point.

Example execution of your program is shown:
alex@linux-PC:/home/Documents$./hex.out
7a
z
B
alex@linux-PC:/home/Documents$./hex.out
30
0
g
alex@linux-PC:/home/Documents$

When you have finished, create a tarball or zip containing only your source code and upload it to Fitchfork on the CS website, using the Practical 3 Task 2 upload link. You have 5 uploads for this task. Name your assembler file **hex.asm**.

### 1.4 Assignment component - 40 Marks [75%]

For the assignment component you are required to implement the basic calculator such that it meets the following requirements:

1. Accepts two positive 7 digit base 10 operands and a operation
2. The four basic numerical operations are to be implemented in their integer form
3. The output is to be displayed as fifteen base decimal (base 10) digits with a sign indicated

Please be advised that this assignment is lengthy and more difficult that initial inspection would lead you to believe. Start well in advance.

Example execution of your program is shown:

alex@linux-PC:/home/Documents$./calc.out
0000005 * 0000050
+00000000000250
alex@linux-PC:/home/Documents$./calc.out
0000005 - 0000050
-00000000000045
alex@linux-PC:/home/Documents$./calc.out
0000055 / 0000007
+000000000000007 r 6
alex@linux-PC:/home/Documents$

### 1.4.1 Assignment Notes

The input values to the program must always be formatted in the same manner. The input format is as follows:

```
[7 digit operand] [space] [operation] [space] [7 digit operand] [space] [newline]
```
The division operation should yield the quotient as a whole number as well as a remainder (even if the remainder is zero).

When you have finished, create a tarball or zip containing only your source code and upload it to Fitchfork on the CS website, using the Assignment 3. You have 20 uploads for this task. Name your assembler file `calc.asm`.

## 2 Mark Distribution

<table>
<thead>
<tr>
<th>Activity</th>
<th>Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td>5</td>
</tr>
<tr>
<td>Task 2</td>
<td>20</td>
</tr>
<tr>
<td>Assignment component</td>
<td>75</td>
</tr>
<tr>
<td>Total</td>
<td>100</td>
</tr>
</tbody>
</table>