School of Electrical and Information Engineering
University of the Witwatersrand, Johannesburg
ELEN2004/ELEN2020 - Software Development I

Lab 1

Lab 1 consists of 3 tasks. Each week one task must be completed and submitted (refer to the submission schedule **`elen2004_lab_deadlines.pdf`** available on Ulwazi.) Note that the following test and submission instructions apply for all tasks.

Testing your code

You should refer to the provided set of inputs and outputs listed in the lab brief to determine if your code is running correctly. Given the set of inputs your code should be able to reproduce the output provided. Make sure to test it with additional values not in the lab brief to ensure it is working correctly.

Submission instructions

Your source code (.cpp) must be submitted to the *Lab 1 - task X* link under *Assignments* on Ulwazi (where **`X`** is the task number). **Only submit your source code.**

It is important that you follow **good programming practices** in your code. This includes

- indent code correctly (use the AStyle formatting tool in Code::Blocks),
- add necessary comments and choose meaningful names of variables in your code,
- add the date and author at the top of your source code. Add the following comment at the start of each source code file.

```
1 // Student number:
2 // Date:
```

Task 1 (week 1) - Number conversion

When you purchase a storage device, the capacity is usually listed by the manufacturer in units of GB. This could for example be 16 GB. Manufacturers use 1000 bytes to represent 1 kB, however 1024 bytes is the actual size of 1 kB. A table showing manufacturers' conversions can be seen in Table 1.

A table showing the actual conversions can be seen in Table 2.

Draw a flowchart on paper that converts the manufacturer storage capacities, listed in Table 3, into the actual amount of storage space you can expect to get from a device. You should

Table 1: Table showing manufacturers' conversions.

1 KB	1000 bytes
1 MB	1000 KB
1 GB	1000 MB

Table 2: Table showing correct conversions.

1 KB	1024 bytes
1 MB	1024 KB
1 GB	1024 MB

- first come up with an algorithm for performing the conversion discussed above.
- The algorithm should prompt the user for an input of a manufacturer capacity and
- display the actual capacity.

Check your flow chart is correct by calculating the values yourself and checking what output your flowchart gives.

Translate your flowchart into C++ code. Make sure your code can display numbers correct to 2 decimal places. Your code should accept an input (in the terminal/console) of a given manufacturer capacity in GB, print to the terminal/console the converted actual capacity in GB and terminate.

Using your C++ code complete the table below. Your code will be tested on the inputs provided.

Table 3: Complete the table.

Manufacturer capacity (GB)	Actual capacity (GB)
40	37.25
15.36	14.31
16	14.90
1000	
8	
10	
102.4	
36	
51.2	
1	

Example 1

An example of an input could be the number **40** followed by the 'enter' key. Your console must then produce the following output and terminate

Listing 1: Output 1

```
37.25
```

Example 2

Another example of an input could be the number **15.36** followed by the 'enter' key. Your console must then produce the following output and terminate

Listing 2: Output 2

```
14.31
```

Hints: Pay attention to what data type you should be storing the user input as. Also make sure to use the **setprecision** and **fixed** keywords to correctly display two decimals in your output.

Task 2 (week 2) - Even or Odd

Write a program that determines whether a number is even or odd. Your program must do the following

- accept an input from a user using the terminal/console,
- check whether this number is even or odd and
- display **even** if the number is even or **odd** if the number is odd.

First draw a flow chart to plan your logic and then attempt the code.

Example 1

An example of an input could be the number **15** followed by the 'enter' key. Your console must then produce the following output and terminate

```
odd
```

Example 2

Another example of an input could be the number **158** followed by the 'enter' key. Your console must then produce the following output and terminate

```
even
```

Task 3 (week 3) - Vending machine price checker

Write a program that displays the cost of different options at a vending machine. Your program must

- read in a selection **`char`** from a file called **`input.txt`**,
- use case statements to select the price linked to the input **`char`**,
- write the price corresponding to the **`char`** to a file called **`output.txt`**.
- If the **`char`** is not present in the file write **`-1`** to the file.

You should account for all items as listed in Table 4 as well as invalid selections (those not listed in table 4).

First draw a flow chart to plan your logic and then attempt the code.

Example 1

An example of what the input and output files contain can be seen below in Listings 3 and 4.

Listing 3: Input example.

```
a
```

Listing 4: Output example.

```
R5.50
```

Example 2

A second example of the input and output files can be seen below in Listings 5 and 6.

Listing 5: Input example.

```
p
```

Listing 6: Output example.

```
-1
```

Table 4: Vending machine options.

Product	Selection **`char`**	Cost
Cheese naks	**`a`**	R5.50
Doritos	**`b`**	R7.50
Coke	**`c`**	R12.00
Chappie	**`d`**	R2.00
Tex	**`e`**	R8.00
Bar-one	**`f`**	R7.00