CSE 102 Programming Assignment 5

DUE

[redacted]

Description

- This is an individual assignment. Please do not collaborate
- If you think that this document does not clearly describes the assignment, ask questions before its too late.

For this assignment, you are expected to write an interactive program which, according to the user input, creates dynamic responses and do simple calculations.

A compound object will be defined interactively and you program will keep track of a simple property and return a result based on the composition of the object.

Example:

Suppose that you want to calculate the total cost of a bicycle. In this case your compound object is a `bicycle`. A bicycle has many parts and some of the parts are also compound objects individually. So, a hierarchical structure can be used in order to model such a compound object. You can visualize it as a tree. Nodes are the parts and the children of a particular node are the parts which make up the part associated with that node.

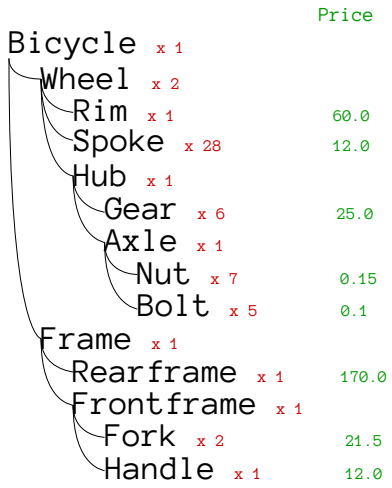

Figure 1: Composition of a bicycle

`Figure 1` shows parts of a bicycle. A bicycle can be described as an object which consists of 2 wheels and a frame. We can create a more detailed description by including the description of the individual parts, `wheel` and `frame`. A description includes number of the parts, part name and, if no further description is needed, prices of the part or parts.

Your program does not know anything about the compound object. Expectation is either there is a compound object which can be specified by its parts or there is a simple object which can be specified by its `property`

For this example, the `property` is the price of an object. Your program will ask questions to the user and eventually calculate the total price of the bicycle the user described.

Below is a simple dialog which the user describes the object 'bicycle' to the program:

```
> Define the object:
> Bicycle
```

```
> What is Bicycle?:
> 2*Wheel+1*Frame
> What is Wheel?:
> 1*Rim+28*Spoke+1*Hub
> What is Rim?:
> 60.0
> What is Spoke?:
> 12.0
> What is Hub?:
> 6*Gear+1*Axle
> What is Gear?:
> 25.0
> What is Axle?:
> 7*Nut+5*Bolt
> What is Nut?:
> 0.15
> What is Bolt?:
> 0.1
> What is Frame?:
> 1*Rearframe+1*Frontframe
> What is Rearframe?:
> 170.0
> What is Frontframe?
> 1*Fork+1*Handle
> What is Fork?:
> 21.5
> What is Handle?:
> 12.0
> Total cost of Bicycle is 1320.1
```

Remarks

- You will construct the required data structure on-the-fly.
- You can assume the maximum length of a user input will not exceed `100` characters.
- There may be spaces in between the terms of the user input. Below is a valid input:

  ```
  1*       Rearframe   +   1    *   Frontframe
  ```

- Do not submit your code without testing it with several different scenarios. `Bicycle` and its structure are presented as an example only. Your program should run if the user tries to describe another object. For example: A car engine, a meal, etc...
- You can use c structs, unions, arrays, c strings, pointers, dynamic memory allocation, etc...
- Write comments in your code.
- If your code does not compile you will get `0`
- Do not share your code with your classmates.

Turn in:

- Source code of a complete C program. Name of the file should be in this format: `<full_name>_<id>.c`.
- Example: `gokhan_kaya_000000.c`. Please do not use any Turkish special characters.
- You don't need to use an IDE for this assignment. Your code will be compiled and run in a command window.
- Your code will be compiled and tested on a Linux machine(Ubuntu). GCC will be used.
- Make sure that your program does not require specific encodings/markings/line-ending-chars. Make sure it works with a file created in a linux environment.
- Make sure you don't get compile errors when you issue this command : `gcc <full_name>_<id>.c`.

- A script will be used in order to check the correctness of your results. So, be careful not to violate the expected output format.
- Provide comments unless you are not interested in partial credit. (If I cannot easily understand your design, you may loose points.)
- You may not get full credit if your implementation contradicts with the statements in this document.
- If your program requires additional compile and link options, state that requirement at beginning of your source code as a comment.

Late Submission

- Not accepted.

Grading (Tentative)

- `Max Grade : 100`.
- Multiple tests(at least `5`) will be performed.

All of the followings are possible deductions from `Max Grade`.

- `#define HARD_CODED_VALUES -10`. (Do **NOT** use hard-coded values)
- No submission: `-100`. (be consistent in doing this and your overall grade will converge to `N/A`) (To be specific: if you miss `3` assignments you'll get `N/A`)
- Compile errors: `-100`.
- Irrelevant code: `-100`.
- Major parts are missing: `-100`.
- Unnecessarily long code: `-30`.
- Inefficient implementation: `-20`.
- Using language elements and libraries which are not allowed: `-100`.
- Not caring about the structure and efficiency: `-30`. (avoid using hard-coded values, avoid hard-to-follow expressions, avoid code repetition, avoid unnecessary loops).
- Significant number of compiler warnings: `-10`.
- Not commented enough: `-5`. (Comments are in English).
- Source code encoding is not `UTF-8` and characters are not properly displayed: `-5`. (You can use 'Visual Studio Code', 'Sublime Text', 'Atom' etc... Check the character encoding of your text editor and set it to `UTF-8`).
- Missing or wrong output values: `Fails the test`.
- Output format is wrong: `-30`.
- Infinite loop: `Fails the test`.
- Segmentation fault: `Fails the test`.
- Fails 5 or more random tests: `-100`.
- Fails the test: `deduction up to 20`.
- Prints anything extra: `-30`.
- Requires space/newline at the end of the file: `-20`.
- Requires specific newline marking (CR/LF): `-20`.
- Unwanted chars and spaces in output: `-30`.

- Submission includes files other than the expected: `-10`.
- Submission does not follow the file naming convention: `-10`.
- Sharing or inheriting code: `-200`.