Carl Underkoffler
Erik Holbroock
CSCI 1300

Project: Identifying Valid Reactions

Introduction:

The goal of this project was to make organic synthesis questions easier by suggesting a multitude of valid reactions for a certain compound. It achieved this by taking in a compound's name, recognizing the position of the active groups and carbon backbones on the compound, and then writing the possible reactions for this compound to a text file.

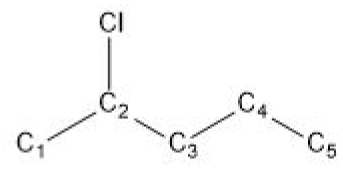
Choice of Input:

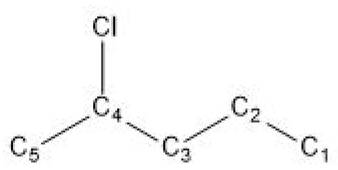
Ideally the structure of the compound would be given to the program as a drawn picture, and the program could store all the details of the compound from the picture. This would be very simple because compounds are easier to draw than name. However, the only two inputs I can work with are strings and numbers. This leaves inputting the compound's name as the best means of interpreting the compound. Since names are a single line of text, it can be difficult to create a system that describes a three dimensional molecule. Although a concrete, accepted naming system already exists, it was changed so that it wouldn't be very hard to create a name after seeing the picture of the compound. In the true naming system, every compound as only one name. For the purpose of this program, the rules associated with making this requirement were dropped in order to make naming compounds simpler.

Explanation of Backbones and Groups:

In organic chemistry, most compounds have a backbone of carbon atoms to which active groups attach. The carbon backbones are like hallways in a hotel, the carbons themselves are like a room, and groups are like the people staying at the hotel. The names of the backbone described the orientation and size of the backbone. The name of the backbone (hallway) is surrounded by angled brackets in my system. When naming a compound, the carbons on a backbone are each given a number (like the number of a hotel room) so that it can represent the location of the group. This number is put in front of the group to which it refers. The group is then put in front of the backbone which it is on. In the real naming system, the numbering of the carbons is very important to make sure one compound has one name. This is not need for this program though.

Eg:
2-chloro-<pentane> or 4-chloro-<pentane>



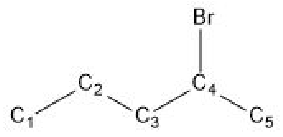


Both names describe the same exact compound. From the name we know that the Chloride atom is bonded to the second (or fourth) carbon. From the backbone within the brackets, "pent" means that the backbone has 5 carbons. The "ane" means that the carbons are orientated in a chain.

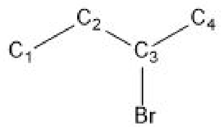
Carl Underkoffler
Erik Holbroock
CSCI 1300

The tricky part comes when multiple backbones are attached to other backbones. In the naming standard for this project, a number must be given to represent the location of the bond on one backbone and another number for the location on the second backbone. Due to the nature of a name trying to convey a structure, naming a compound is like standing in a certain hallway in a hotel and then describing everything that comes off that primary hallway. This primary hallway is the primary backbone. In the name of the compound, the location of a secondary backbone is followed by the information about the secondary backbone in parenthesis. At the beginning of the parenthesis is the location of the primary backbone relative to the secondary enclosed in curly brackets.

Eg:

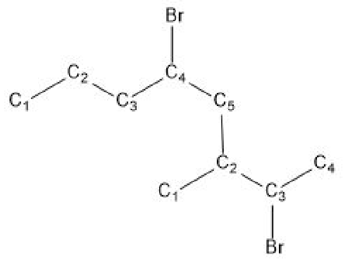


4-bromo-<pentane>

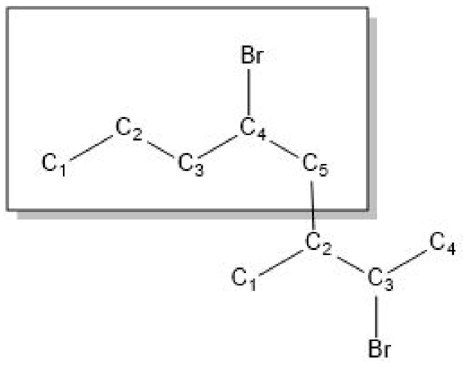


3-bromo-<butane>

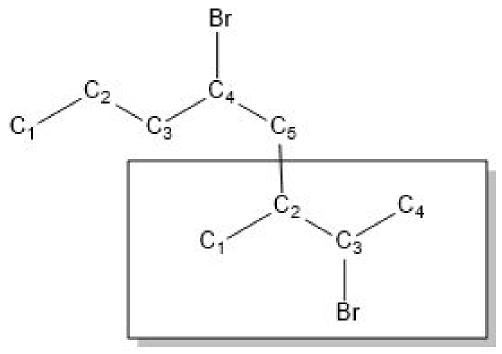
Attach these two carbons to form a compound with multiple backbones:



Naming the combined backbone by choosing a different primary backbone each time:



5-({2}-3-bromo-<butane>)-4-bromo-<pentane>



2-({5}-4-bromo-<pentane>)-3-bromo-<butane>

Notice the secondary backbone is in parenthesis, and the attachment point to the secondary backbone is in curly brackets.

allNames Class:

Thinking ahead, the best way to examine each carbon is to cycle through them all in a list. This could be done by representing the backbone as vector and its attachments the values stored in it. To find out the attachments, a simple program could be made to fill the vector based off the primary compound of carbon name. Then, rather than creating a different method for the other levels of backbones, create an algorithm that renames the compound with a new primary backbone. In the allNames class, this done by the function invert_names() function. It works by taking out the first secondary compound and add the original backbone to it as if it was a secondary backbone. The limits of this method, are that compound must only contain three backbones, and they must originally be named from one end of the chain of backbones. This is because if the compound has two secondary backbones to start with, the second time it is renamed, the backbones will be improperly attributed. I then tried to make a different algorithm that would simply create each name from scratch based off the level depth the carbon was at and the ones that surrounded it. Since I could not get this method to work, I returned to the original method with its limitations. The limitations are acceptable because most of the compounds that I would see could be named using three backbones or less. Within the allNames class, all the viable names have all non-primary groups and non-secondary backbones stripped from them. This was done to make interpreting the name easier, and the only loss of functionality was for reactions that I had not wanted to include due to too much complexity, as it is. The only reactions I was concerned with are ones that can be identified by looking at what is attached to a singular carbon, not how groups over multiple backbones are oriented to each other.

Backbone Class:

With the names striped, they are individually passed to the backbone class. The backbone class takes out the actual name of the backbone, and makes a vector (the carbon vector) of appropriate size by set_bbname(). Then the carbons vector is populated by the Carbon-Carbons bonds in the backbone by populate_carbons(). The type of backbone will change how the carbons vector is populated, which is why the types of backbones must be hard written into the program and cannot just be included as a valid backbones list in a text file. Since the backbone is a variable size, the carbons vector is of variable size. This means that it would not be possible to make an indefinite amount of vectors where each item was an individual group. This lead me to just add groups to a carbon by extending the string with a space, which is sloppy but works.

The groups on the backbone are placed in the carbons vector by the set_carbons() function. This is done by merely searching through the name for valid tags (essentially valid groups) and then inserting them into carbons vector. This method means that anything could be inserted as a group but it will only be added to the carbons vector if it is valid. The only bad thing about this is I do not have any message saying that there is an unrecognized group.

The load_matrix() function is used so that the carbons vector can be copied to an external carbons vector in the main function. This external carbons vector can then be sent to the reactions class for it to be check by the reaction requirements.

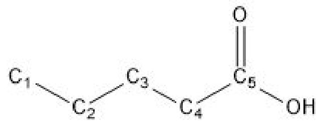
Reaction Class:

The reaction class loads in the external carbons vector and checks it by each reaction requirement. The reactions themselves are inputted via a text file and their requirements are written in such a way so that the class can understand them effectively. The carbons group must have all the requirements and none of the blocks. It will store the reaction as a valid one if either the forward or reverse requirements are met. Because this class can receive only one backbone of a compound at a time, it is constructed prior to receiving any backbones and stores all the valid reactions throughout each backbones input. This allows for a singular output to be written to the output text file at the end of the program.

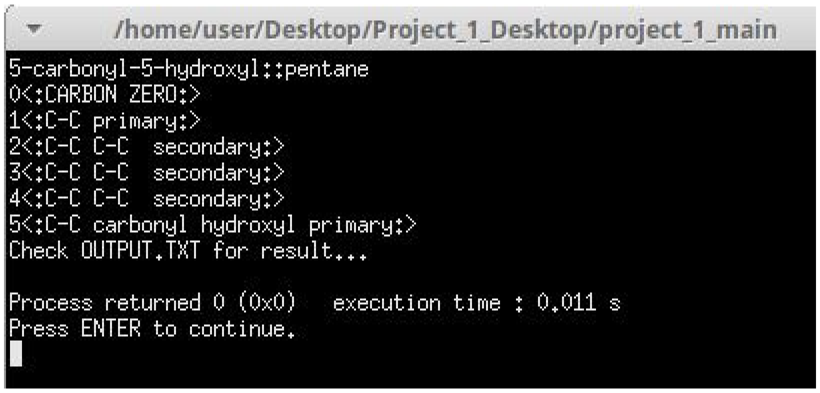
Test Cases:

Console output will be all the stripped names, and the contents of the carbon vector at each index. The carbon vector pertains to the primary backbone in the stripped name above. The primary backbone follows the double colons. Keep in mind, the stripped names only keep information about what connects to that primary backbone. The output.txt should validate that the proper reactions were printed.

5-carbonyl-5-hydroxyl<pentane>



Carboxylic Acid -> Program should put primary, carbonyl and hydroxyl on the fifth carbon in carbons vector. There is only one backbone, so there should only be one stripped name.

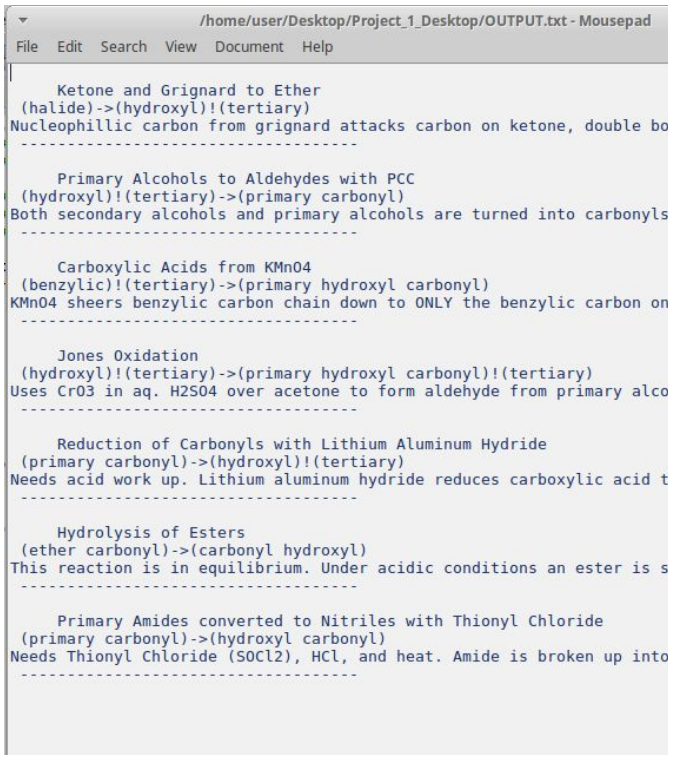


```
5-carbonyl-5-hydroxyl::pentane
0<:CARBON ZERO:>
1<:C-C primary:>
2<:C-C C-C  secondary:>
3<:C-C C-C  secondary:>
4<:C-C C-C  secondary:>
5<:C-C carbonyl hydroxyl primary:>
Check OUTPUT.TXT for result...

Process returned 0 (0x0)   execution time : 0.011 s
Press ENTER to continue.
```

Reactions:

Carl Underkoffler
Erik Holbroock
CSCI 1300

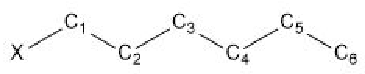
“Ketone and…” -> Should print because its hydroxyl is primary
“Primary Alc..” -> Has primary alcohol and primary carbonyl
“Carboxylic Acid..” -> Is Carboxylic Acid
“Jones oxi…” -> Is primary hydroxyl carbonyl (carboxylic acid)
“Reduction of…” -> Is primary carbonyl (carboxylic acid)
“Hydrolysis…” -> Not ether carbonyl but is carbonyl hydroxyl
“Primary Amides” -> Is hydroxyl carbonyl


/home/user/Desktop/Project_1_Desktop/OUTPUT.txt - Mousepad
File Edit Search View Document Help

```
    Ketone and Grignard to Ether
 (halide)->(hydroxyl)!(tertiary)
Nucleophillic carbon from grignard attacks carbon on ketone, double bo
 ------------------------------------

    Primary Alcohols to Aldehydes with PCC
 (hydroxyl)!(tertiary)->(primary carbonyl)
Both secondary alcohols and primary alcohols are turned into carbonyls
 ------------------------------------

    Carboxylic Acids from KMnO4
 (benzylic)!(tertiary)->(primary hydroxyl carbonyl)
KMnO4 sheers benzylic carbon chain down to ONLY the benzylic carbon on
 ------------------------------------

    Jones Oxidation
 (hydroxyl)!(tertiary)->(primary hydroxyl carbonyl)!(tertiary)
Uses CrO3 in aq. H2SO4 over acetone to form aldehyde from primary alco
 ------------------------------------

    Reduction of Carbonyls with Lithium Aluminum Hydride
 (primary carbonyl)->(hydroxyl)!(tertiary)
Needs acid work up. Lithium aluminum hydride reduces carboxylic acid t
 ------------------------------------

    Hydrolysis of Esters
 (ether carbonyl)->(carbonyl hydroxyl)
This reaction is in equilibrium. Under acidic conditions an ester is s
 ------------------------------------

    Primary Amides converted to Nitriles with Thionyl Chloride
 (primary carbonyl)->(hydroxyl carbonyl)
Needs Thionyl Chloride (SOCl2), HCl, and heat. Amide is broken up into
 ------------------------------------
```

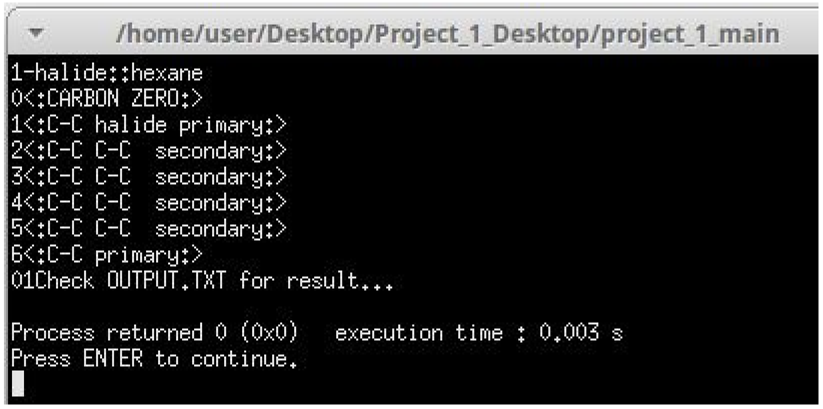
Groups properly made and reactions properly identified.

1-halide<hexane>



Should only appear for halide reactions. Halide is on primary carbon (first carbon). Only one backbone so there should only be on stripped name.

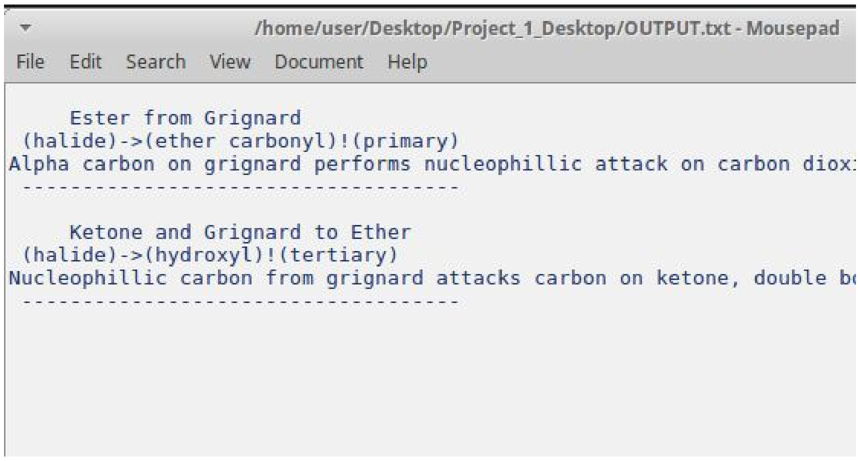
Carl Underkoffler
Erik Holbroock
CSCI 1300



/home/user/Desktop/Project_1_Desktop/project_1_main

```
1-halide::hexane
0<:CARBON ZERO:>
1<:C-C halide primary:>
2<:C-C C-C  secondary:>
3<:C-C C-C  secondary:>
4<:C-C C-C  secondary:>
5<:C-C C-C  secondary:>
6<:C-C primary:>
01Check OUTPUT.TXT for result...

Process returned 0 (0x0)   execution time : 0.003 s
Press ENTER to continue.
```

Valid reactions:
“Ester from…” -> halide
“Keton and... “ -> halide



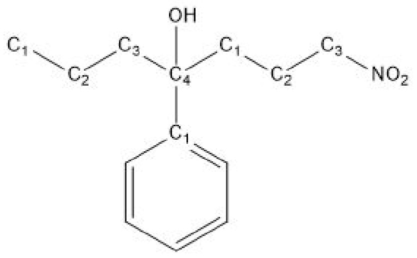
/home/user/Desktop/Project_1_Desktop/OUTPUT.txt - Mousepad

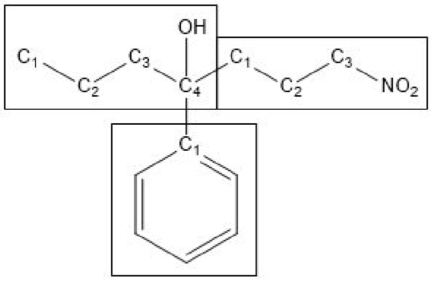
```
    Ester from Grignard
 (halide)->(ether carbonyl)!(primary)
Alpha carbon on grignard performs nucleophillic attack on carbon diox
 -----------------------------------

    Ketone and Grignard to Ether
 (halide)->(hydroxyl)!(tertiary)
Nucleophillic carbon from grignard attacks carbon on ketone, double bo
 -----------------------------------
```

Output shows valid reactions only. Groups properly assigned.

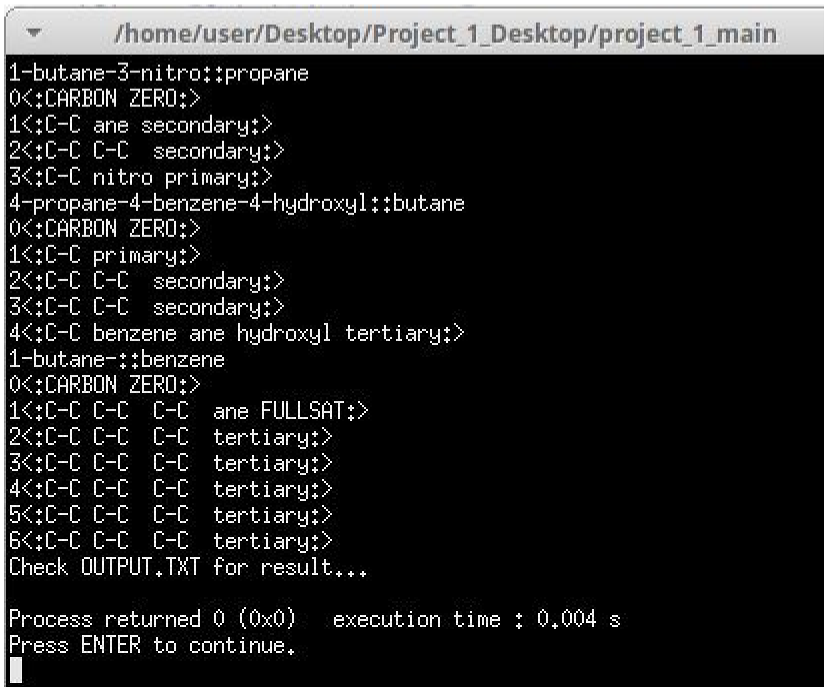
1-({4} 4-(<benzene>)-4-hydroxyl<butane>)-3-nitro<propane>

In the copy of the molecule I boxed off the three backbones so that their connectivity can be seen a little better. The four carbon chain on the left is the butane, the three carbon on the right is the propane, and the ring like structure is the benzene.





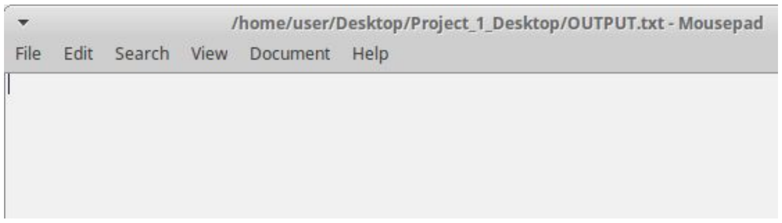
When propane is the primary backbone, I expect its carbons vector to have a nitro group on its third carbon and an “ane” from the “butane” on its first carbon.
When butane is the primary backbone, I expect its carbons vector to have a “benzene”, “ane” from the “propane”, and “hydroxyl” on its fourth carbon.
When benzene is the primary backbone, I expect its carbons vector to have an “ane” from the “butane” on it first carbon.



/home/user/Desktop/Project_1_Desktop/project_1_main

```
1-butane-3-nitro::propane
0<:CARBON ZERO:>
1<:C-C ane secondary:>
2<:C-C C-C  secondary:>
3<:C-C nitro primary:>
4-propane-4-benzene-4-hydroxyl::butane
0<:CARBON ZERO:>
1<:C-C primary:>
2<:C-C C-C  secondary:>
3<:C-C C-C  secondary:>
4<:C-C benzene ane hydroxyl tertiary:>
1-butane-::benzene
0<:CARBON ZERO:>
1<:C-C C-C  C-C  ane FULLSAT:>
2<:C-C C-C  C-C  tertiary:>
3<:C-C C-C  C-C  tertiary:>
4<:C-C C-C  C-C  tertiary:>
5<:C-C C-C  C-C  tertiary:>
6<:C-C C-C  C-C  tertiary:>
Check OUTPUT.TXT for result...

Process returned 0 (0x0)   execution time : 0.004 s
Press ENTER to continue.
```

Valid Reactions:
NONE



/home/user/Desktop/Project_1_Desktop/OUTPUT.txt - Mousepad

All the stripped names are correct. There are no reactions written that use the nitro group yet, so the program does not recognize any valid reactions but it is still a valid tag (group). Notice the hydroxyl group prompts zero reactions because of the “tertiary” block on all the hydroxyl reactions. This tertiary tag is the reason why all the names needed to be inverted so that each backbone could be examined as the primary backbone. Notice the hydroxyl group did not trigger the “carbonyl hydroxyl” requirement in the “Hydrolysis of Esters” reaction. This is because all requirements must be met and there is no carbonyl on this compound.

Conclusion:

Altering the conventional naming system was necessary to make creating the user input easy. However, the failure of creating a solid algorithm in the allNames class to create all possible valid names, meant some functionality was cut out of multiple backbone compounds. Also, little features have died along the way in development; the code remains, but their functionality is not carried through to the output. However, for the majority of compounds that would be seen in an organic chemistry class, the program is effective means of identifying valid reactions that could be useful in a synthesis problem.