CS1020E: Data Structures and Algorithms I

Tutorial 2 – Basic Object-Oriented Programming

(Week starting 2 February 2015)

1. Old McDonald

When you were just a little kids, you might have heard the nursery rhymes called "Old McDonald". Its lyric teaches kids about the sounds that animals would make. The most common part of the lyric is about the Cow:

Old McDonald had a farm, E-I-E-I-O
and on his farm he had a ***Cow****, E-I-E-I-O*
with a ***Moo Moo*** *here and a* ***Moo Moo*** *there*
here a ***Moo****, there a* ***Moo****, everywhere a* ***Moo Moo***
Old McDonald had a farm, E-I-E-I-O

The lyric above can be generalized to have *animal names* with *animal sounds*. Some of the other animals used in the lyric are <*Dog, Woof*> and <*Duck, Quack*>. As a computer scientist, you want to code a generic Farm Song to sing Old McDonald. Use the following template to solve your problem.

```
class OldMcDonald {
  private:
    Animal** _farm; // Old McDonald had a farm (still has now)
    int _size;      // Farm size
  public:
    /* Constructor and Adding Animals are omitted for brevity */
    void sing() {
      for(int i=0; i<_size; i++) {
        cout << "Old McDonald had a farm, E-I-E-I-O" << endl;
        // Add the rest of the lyrics here...
      }
    }
};
class Animal {
      // TODO
};
```

To show that your program works, add the following animals to the list: <*Cow, Moo*>, <*Dog, Woof*>, and <*Duck, Quack*>.

2. Irregular Input Pattern

Normally, people's names can be categorized into *first name* and *last name*. However, some people have a name with multiple middle names (one of our tutors, Adi Yoga Sidi Prabawa is one such example). Given the irregularity of the number of middle names, having a fixed format for input (assuming standard input) such as *first_name last_name* is not exactly feasible. If we are only interested in names, then we can always read until the end of line and parse the number of whitespaces.

To complicate matters, let us assume a use case of personal data which includes names, age, and gender. A programmer designs an ingenious method of encoding the possibility of multiple middle names into a single line input. In fact, the entire inputs can be written in a single line. The input format is the following:

$first\_name_1$ $[middle\_name_1]^*$ $last\_name_1$ # age_1 $gender_1$
$first\_name_2$ $[middle\_name_2]^*$ $last\_name_2$ # age_2 $gender_2$
...

An explanation of the input is this:
1. Every person always has a first name and a last name.
2. Every person may have 0 or more middle names.
3. The name (the full name) ends after the input reaches # then it is followed by the age and gender directly.
4. The next part (if any) will be part of the next input.

Notice that the entire input can be written into a single line because age and gender would not have whitespace in the middle (i.e. it is one-word). Thus, you are *not allowed* to read the entire line (e.q. using `getline`). Your job is to write a program to read the input and for each input, create the following `struct`. Use `cin` for your input.

```
struct Student {
      string first;
      string middle; // Combine all the middle names
      string last;
      int age;
      bool female;   // true if female, false if male
};
```

3. Overloading

Given the following five functions:

```
int add(int a, int b) { return a + b; }             // 1
double add(double a, double b) { return a + b; } // 2
double add(int a, int b) { return a + b; }         // 3

void swap(int a, int b) {     // 4
  a = a ^ b;
  b = a ^ b;
  a = a ^ b; }
void swap(int& a, int& b) {  // 5
  a = a ^ b;
  b = a ^ b;
  a = a ^ b; }
```

a) Which functions are having conflict in the overloading?
b) What causes those overloading errors?
c) Removing the *offending* function, would the following statement cause error?
if yes, how would you correct the error (besides removing the offending function call)?
if no, which function is called in each statement?
`add(1, 2); add(3.0, 4); add(5, 6.0); add(7.0, 8.0);`

CS1020E: Data Structures and Algorithms I

4. String Manipulation. (Virus)

Every Website has a domain and some content. A website is infected if its content contains at least one virus pattern. Input an integer N<=100 followed by N websites. Every website is in form of its domain followed by its content (no white spaces). Then input an M<=100 followed by M virus patterns. You should output the domain of all the infected websites.

```
#include <iostream>
#include <string>
using namespace std;

class Website {
    string a_domain;
    string a_content;
public:
    Website() {};
    /** More Methods **/
};

int main(){
  int n,m;

  cin>>n;
  Website * websites = new Website[n];
  /***Your Codes ***/
   return 0;
}
```