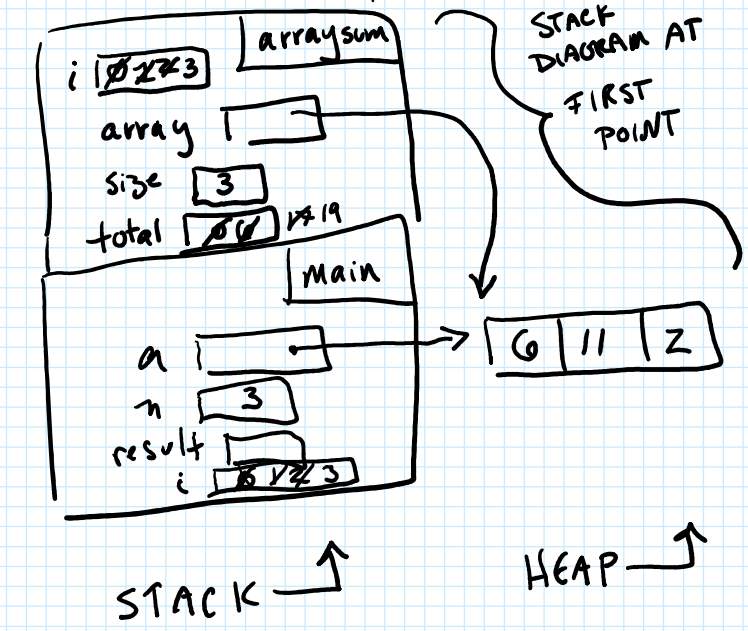
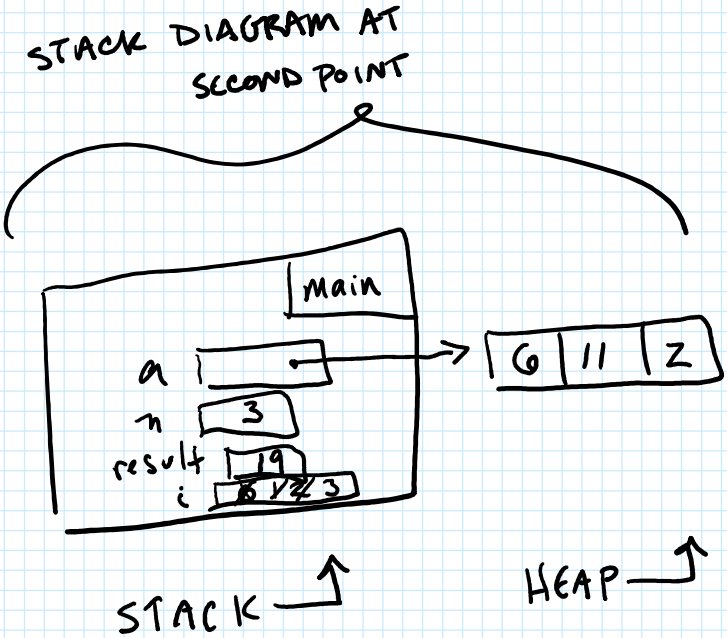
2.1 classes and objects in C++

Monday, September 5, 2022 22:42

Reminders:
- lab 1 is due Wednesday evening (git add, git commit, git push)
- code from lecture is available at:
  /home/fontes/public/cs35/

Another stack example: stack_example_dynamic.cpp





BIG CS 35 IDEA:

There are lots of different ways to implement a data structure.

INTERFACE: how you communicate & interact with the data structure

IMPLEMENTATION: how it works behind the scenes

C++ classes and objects

OBJECTS combine { data (attributes and data members)
functionality (methods)

CLASS describes a type of object.

An OBJECT is a particular instance of a class.

example: a Student class

data members: age, grade level, class year, name, GPA

methods: changeAge, setGPA, changeClassYear, howLongUntilGraduation, getName

Q: Why does C++ organize things this way?

So that data can be protected.

SYNTAX for declaring a class in C++:

```
class <class name> {
    private: // usually all data goes here
        <type> <variable name>;
```

```
    <type>  <variable name>;
      ⋮
  public: // usually all your methods go here
    <type> <method name> (<type> <parameter name>, ...);
    <type> <method name> (<type> <parameter name>, ...);
      ⋮
};   <-- DON'T FORGET THIS SEMICOLON!!
```

SYNTAX for defining a method of a class in C++:

```
<return type> <class name> :: <method name> (<type> <param name>, ...) {
    <body of function>
}
```

This is important to specify it's a method of this class.

Each method gets defined like this.

example: Point

idea: This object is an (x,y) point.

data members

```
float x;
float y;
```

methods

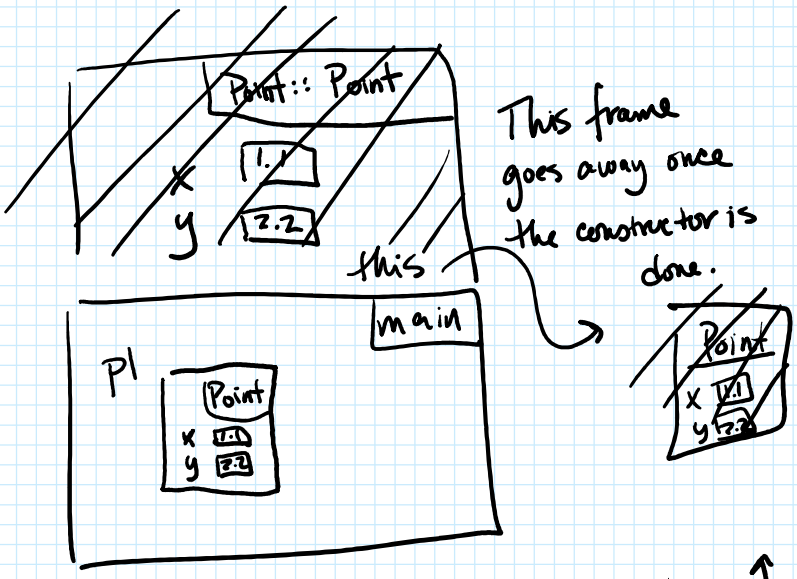
```
void Print();
float GetDistFromOrigin();

// getters
float getX();
float getY();
// setters
...setX(float a);
```

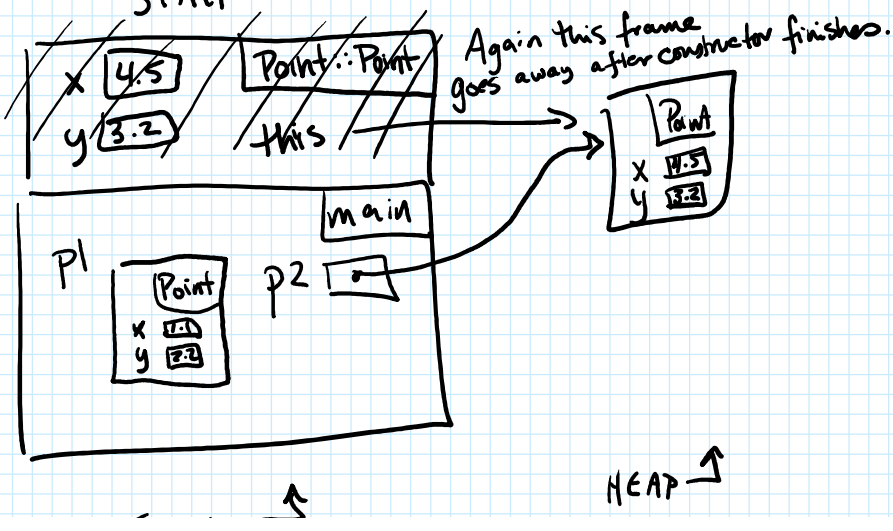
```
// setters
void setX(float a);
void setY(float a);

// constructor
Point(float x, float y);
```

no return type on constructor, name is same as name of class

We can declare a class
define its methods
use the class
} all in one file.

See file: point.cpp





STACK ↑ HEAP ↑

STACK DIAGRAMS with OBJECTS

- as before: static objects are stored in the stack
  dynamic objects are stored in the heap
- all the memory for an object should be drawn in a box labelled with the class name
- every method creates a new stack frame

every method frame automatically includes the "this" variable which points to the memory block for the object

It is possible to separate the 3 parts into 3 files --- see point_version2

1. <class name>.h
   This file provides an INTERFACE, and declares the class.
2. <class name>.cpp
   This file contains the definition of the class methods, and must contain the line:
   #included "point.h"
3. main program
   This file uses the class and must contain the line:
   #include "point.h"

To compile this will take 3 steps. (We'll soon automate this so you don't have to do so much typing.)

1. clang++ -std=c++11 -c point.cpp -o point.o
2. clang++ -std=c++11 -c main.cpp -o main.o
3. clang++ -std=c++11 main.o point.o -o main

The "-c" lines mean "compile this, but don't create an executable file."
The final step links together all the pieces into a single executable file called "main" (in this example).