Static Data & Dynamic Memory

CS 165 – Object Oriented Software Development

Macbeth – Lesson 7.3

Agenda

- Music Friday
- Opening Prayer
- Dynamic Memory (Team Activity & Checkpoint)
- MoonLander Q&A
- Looking Forward

Music Friday

Let Zion in Her Beauty Rise (Hymn 41)

Let Zion in her beauty rise;
Her light begins to shine.
Ere long her King will rend the skies,
Majestic and divine,
The gospel spreading thru the land,
A people to prepare
To meet the Lord and Enoch's band
Triumphant in the air.

Static Variables

- A static variable is shared memory between objects of the same class
- How would use a static variable in the following circumstances:
  - Define constant variables for all classes to use
  - Keep track of the number of objects you have created
  - Keep track of the number of times a function has been called
  - Provide tuneable parameters – something you can change once and will apply to all objects.

- Declare in your class (.h):

```
class Fault
{
   private:
      static int faultConfirmationTime;
   public:
      void setConfirmationTime(int time); // will do a faultConfirmationTime = time;
}
```

- Initialize (.cpp):

```
int Fault::faultConfirmationTime = 5;
```

Pointers

- Pointers are variables that contain addresses to other variables (any kind .. even pointers)
- Declaration:
    - Variable of type int: `int a`
    - Pointer to a variable of type int: `int *e`
- Expressions:
    - Address of a variable: `&a`
    - Value at an address: `*e`
- Questions:
    - What are the expressions to set pointers e, `f`, g and h?
    - What are the expressions to get value at pointers `e, f, g, and h?`
    - What are the expressions to swap what pointers e and `f` point to?

Variable	**Address**	**Value**
int a	0x1000	123
int b	0x1004	75
int c	0x1008	-42
int d	0x100C	9989
int *e	0x1010	0x1000
int *f	0x1014	0x1004
int *g	0x1018	0x1008
int *h	0x101C	0x100c

Array of Pointers

- You can dynamically create any variable on the heap using the `new` command.
- The `new` command returns a pointer to the data you created (or allocated) on the heap.
- You use the pointer to access the data and remove (or deallocate) it using the `delete` command.

- In these examples, we are creating an array.
  - Create an array of integers
  - Create an array of pointers to integers

- How can these arrays be accessed using for loops and:
  - Using the [#] notation; or
  - Using pointer arithmetic

int *data = new int[4]

Index	**Value**	**Address**
0	123	0x1000
1	75	0x1004
2	-42	0x1008
3	9989	0x100C

int **dataPtrs = new int*[4]

Index	**Value**	**Address**
0	0x1000	0x1010
1	0x1004	0x1014
2	0x1008	0x1018
3	0x100C	0x101C

Pointers to Objects

- Just like you can create variables (including arrays) dynamically on the heap, you can also create objects of classes dynamically on the heap using the `new` command.

```
Product *radio = new Product("Bluetooth Radio","Has radio and alarm as well.",49.99, 8.5);
Product *bike = new Product("Mountain Bike","Rugged and made to last.",169.95, 50.5);
Product *unknown = new Product();
```

- Instead of using the dot notation to run functions, if you have a pointer to object, then you need to use the arrow notation.

```
float price = radio->getTotalPrice();
bike->display();
```

- Remember the `this` command? It's a pointer to an object. To access member data for an object with a class you use the arrow notation.

```
this->basePrice
```

Review Checkpoint 7B

Passing Pointers to Functions

- Remember the displayAdvertising function I have in the Product class

```
void Product::displayAdvertising()
```

- I want to overload this function to provide the ability to both display the product details and the details of another product that I want to recommend.
  - What will my display function declaration look like?
  - How will I call it?

```
int main()
{
   Product *radio = new Product("Bluetooth Radio",49.99);
   Product *bike = new Product("Mountain Bike",169.95);
   // call display function here
   return 0;
}
```

BYU IDAHO

Looking Forward

- Monday
    - MoonLander Final Project Due
        - If you have the game working according the core requirements, you may modify other code given to add more features.
        - If you add more features, please list out the new features you added in the comments of you makefile
    - Last chance to submit Checkpoint A and B