CS 367 - Introduction to Data Structures
Thursday, September 22, 2016

Homework h2, due 10 pm TOMORROW, Friday, Sept 23th
Homework h3 assigned Monday, Sept 26th
Program p1, due 10pm Sunday, Oct 2nd. (Should by about half way by now)

Assignment questions? Post it on Piazza or Consult with a TA during scheduled hours.

Report any exam conflicts or McBurney exam accommodations before Oct 7th.
Updated Module 3 with links for Exams and instructions for reporting Exam Conflicts

Email your instructor by tomorrow, Friday, 9/23, if you participate in religious observances that might interfere with course requirements. Include your name, UW ID#, date and explanation.

Last Time
Program Handin Instructions (hand in all files each submission)
Exceptions Review
  • throwing
  • handling
  • execution
  • practice with exception handling

Today
Exceptions Review (from last lecture)
  • throws and checked vs. unchecked
  • defining
Java Primitives vs. References Review
Chains of Linked Nodes
  • Listnode class
  • practice with chains of nodes

Next Time
Read: continue Linked Lists
Chains of Linked Nodes
  • more practice with chains of nodes
Java Visibility Modifiers
LinkedList Class
Primitive vs. Reference Types: Assignment

Primitives

assume code is in main()
int x, y, z;
x = 11;
y = x;
z = x;
z = 33;
y = 22;

→ What does each variable contain after the code above executes?

A.) x has y has z has
B.) x has y has z has
C.) x has y has z has

References

assume code is in main()
ArrayList<String> x, y, z;
x = new ArrayList<String>();
y = x;
z = x;
y = new ArrayList<String>();
z.add("Computer");
y.add("Science");

→ What does each ArrayList contain after the code above executes?

A.) x’s ArrayList has y’s ArrayList has z’s ArrayList has
B.) x’s ArrayList has y’s ArrayList has z’s ArrayList has
C.) x’s ArrayList has y’s ArrayList has z’s ArrayList has

→ What do x, y and z contain?

★
Primitive vs. Reference Types: Parameter Passing

Primitives

Given:

```java
void mod1(int x) {
    x = 42;
}
```

Execute code in `main()`:

```java
int x = 11;
int[] y = {11, 22, 33};
mod1(x);
mod1(y[2]);
```

→ What does variable `x` and array `y` in `main` contain after the code above executes?

A.) `x` has `y`'s array has
B.) `x` has `y`'s array has
C.) `x` has `y`'s array has

→ What happens if we call `mod1(y)` in `main`?
Primitive vs. Reference Types: Parameter Passing

References

Given:

```java
void mod2(int[] x) {
    x[0] = 21;
}

void mod3(int[] x) {
    x = new int[x.length];
    x[0] = 42;
}
```

Execute code in `main()`:

```java
int x = 11;
int[] y = {11, 22, 33};
mod2(y);
mod3(y);
```

→ What does variable `x` and array `y` in `main` contain after the code above executes?

A.) `x` has `y`'s array has
B.) `x` has `y`'s array has
C.) `x` has `y`'s array has

→ What happens if we call `mod2(x)` in `main`?
Programmer’s Memory Model for Java

Call Stack
contains

birth
dead

Heap
contains

birth
dead

Static Data
contains

birth
dead
New Data Structure - Chain of Linked Nodes

The Data Structure

Array vs. Chain of Nodes

Goal
Listnode Class

class Listnode<E> {

    private E data;
    private Listnode<E> next;

    public Listnode(E d) {
        this(d, null);
    }

    public Listnode(E d, Listnode<E> n) {
        data = d;
        next = n;
    }

    public E getData() { return data; }
    public Listnode<E> getNext() { return next; }

    public void setData(E d) { data = d; }
    public void setNext(Listnode<E> n) { next = n; }
}
Practice: Using Listnodes

→ Draw a memory diagram corresponding to the given code:

assume code is in main()

Listnode<String> n1 = null;

→ Write the code that results in: