Boone Tison

Problem Set #1

1. public boolean equals(Object obj) {

if (obj == null) return false;

if (obj !instanceof Point) return false;

Point p = (Point) obj;

if (p.dimensions != this.dimensions) return false;

return Arrays.equals(p.coordinates, this.coordinates);

}

1. public boolean equals(Object obj) {

Point p = obj as Point;

if (p == null) return false;

if (p.dimensions != this.dimensions) return false;

return Arrays.equals(p.coordinates, this.coordinates);

}

1. public int fibonacci(int n) {

static int memos[n+1];

static bool init = false;

if (!init) {

for (int i = 0; i <= n; i++)

memos[i] = -1;

init = true;

}

if (nums[n] == -1) {

if (n == 0 || n == 1) memos[n] = n;

else memos[n] = fibonacci(n-1) + fibonacci(n-2);

}

return memos[n];

}

1. public SliceableList<t> binary-operator :: (int beginIndex, int endIndex) {

if (beginIndex >= endIndex) return new SliceableList<T>();

if (endIndex >= this.size) return new SliceableList<T>();

T[] a = new T[endIndex-beginIndex];

int c = 0;

for (int i = beginIndex; i < endIndex; i++) {

a[c] = this.list[i];

c++;

}

return new SliceableList<T>(a);

}

private int size;

private T[] list;

public SliceableList<T>() {

list = new T[0];

size = 0;

}

public SliceableList<T>(T[] arr) {

list = arr;

size = arr.length;

}

}