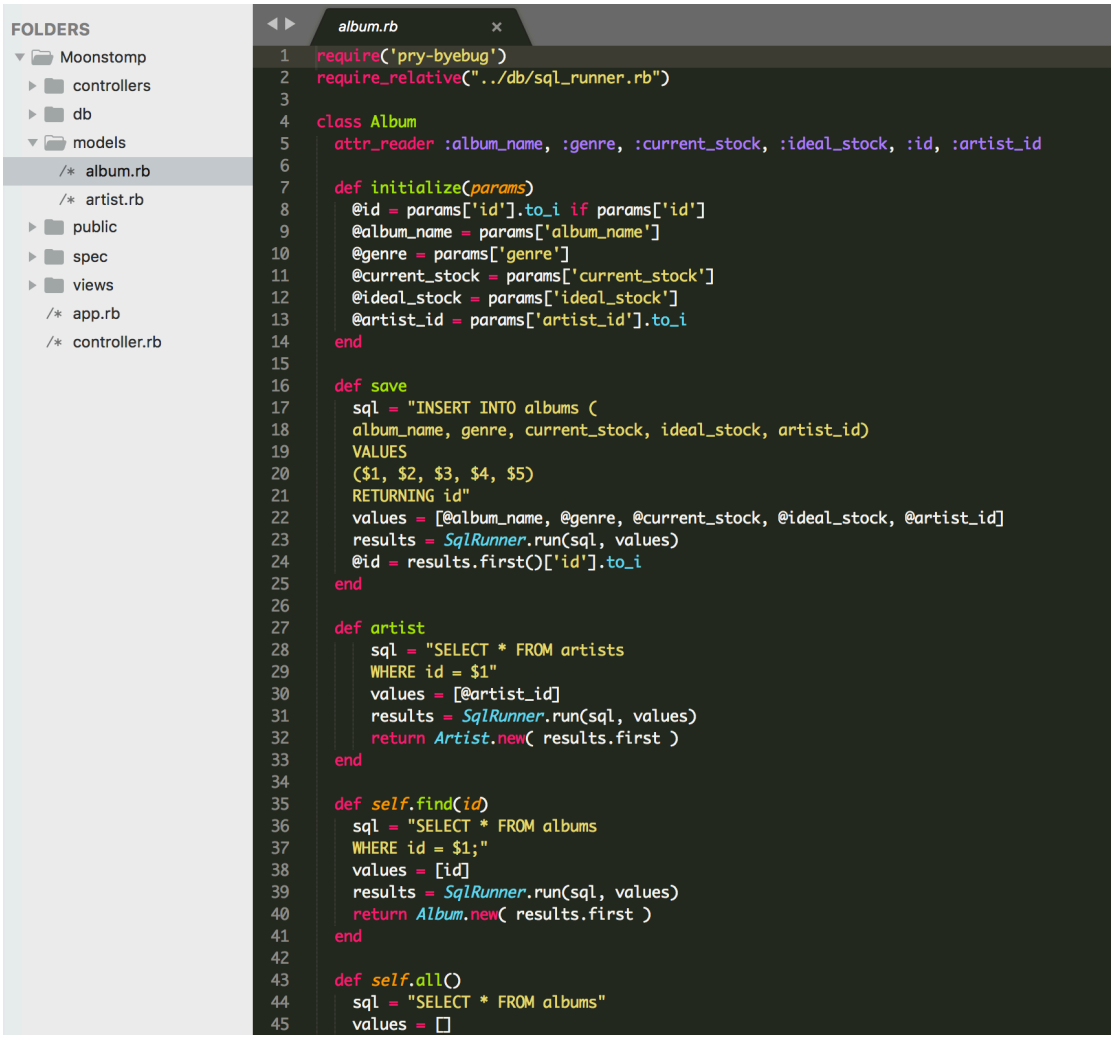
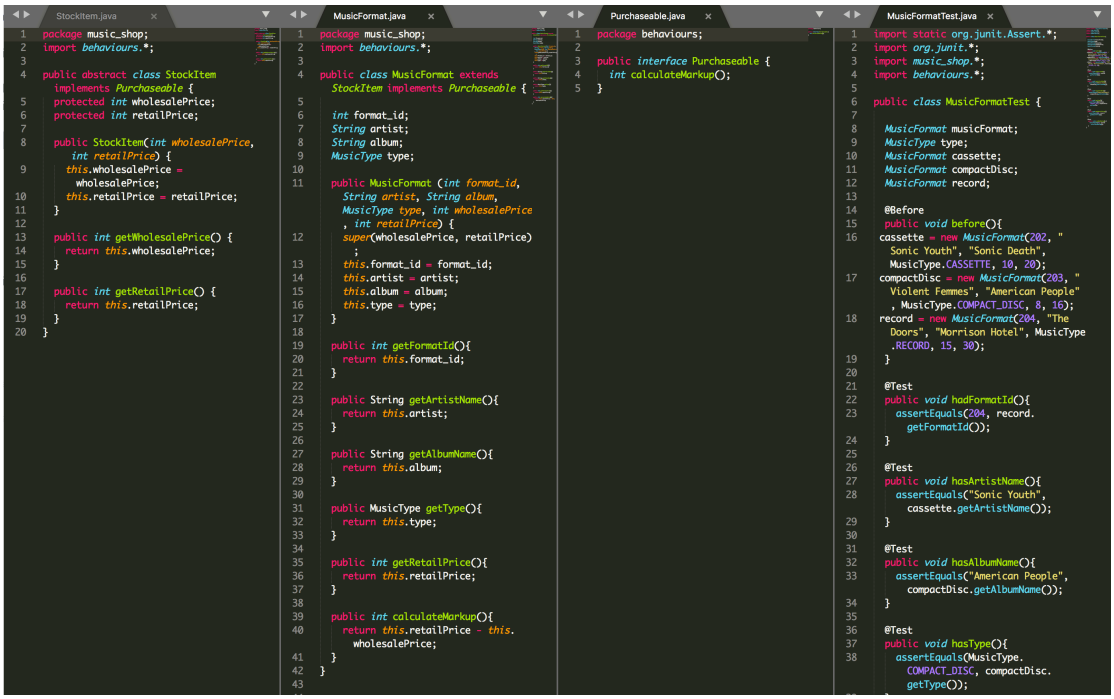
Craig Lynagh
PDA: Software Development

I&T I.T 1



Craig Lynagh
PDA: Software Development

I&T I.T 2



StockItem.java

```
package music_shop;
import behaviours.*;

public abstract class StockItem
  implements Purchaseable {
  protected int wholesalePrice;
  protected int retailPrice;

  public StockItem(int wholesalePrice,
    int retailPrice) {
    this.wholesalePrice =
      wholesalePrice;
    this.retailPrice = retailPrice;
  }

  public int getWholesalePrice() {
    return this.wholesalePrice;
  }

  public int getRetailPrice() {
    return this.retailPrice;
  }
}
```

MusicFormat.java

```
package music_shop;
import behaviours.*;

public class MusicFormat extends
  StockItem implements Purchaseable {

  int format_id;
  String artist;
  String album;
  MusicType type;

  public MusicFormat (int format_id,
    String artist, String album,
    MusicType type, int wholesalePrice
    , int retailPrice) {
    super(wholesalePrice, retailPrice)
      ;
    this.format_id = format_id;
    this.artist = artist;
    this.album = album;
    this.type = type;
  }

  public int getFormatId(){
    return this.format_id;
  }

  public String getArtistName(){
    return this.artist;
  }

  public String getAlbumName(){
    return this.album;
  }

  public MusicType getType(){
    return this.type;
  }

  public int getRetailPrice(){
    return this.retailPrice;
  }

  public int calculateMarkup(){
    return this.retailPrice - this.
      wholesalePrice;
  }
}
```

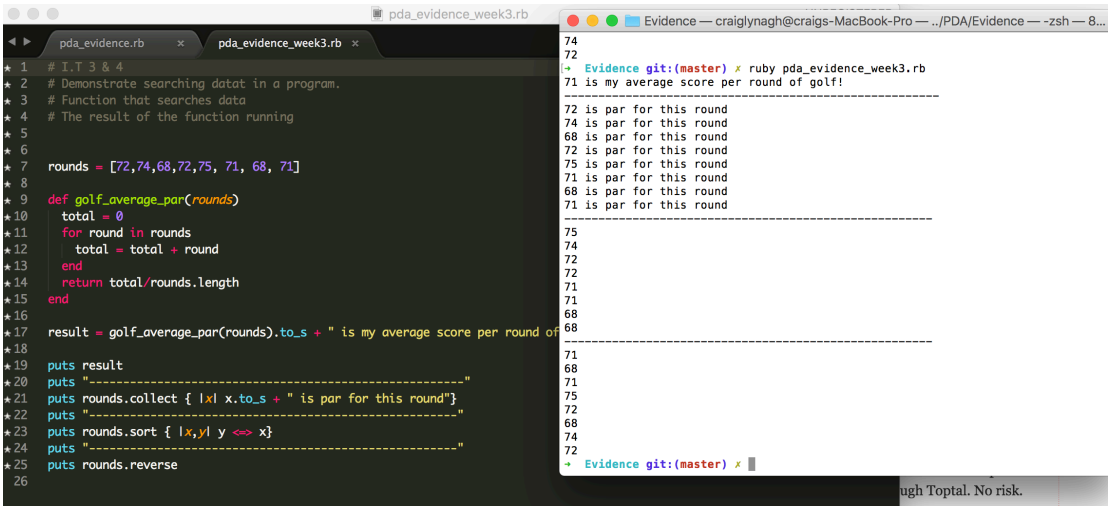
Purchaseable.java

```
package behaviours;

public interface Purchaseable {
  int calculateMarkup();
}
```

MusicFormatTest.java

```
import static org.junit.Assert.*;
import org.junit.*;
import music_shop.*;
import behaviours.*;

public class MusicFormatTest {

  MusicFormat musicFormat;
  MusicType type;
  MusicFormat cassette;
  MusicFormat compactDisc;
  MusicFormat record;

  @Before
  public void before(){
cassette = new MusicFormat(202, "
  Sonic Youth", "Sonic Death",
  MusicType.CASSETTE, 10, 20);
compactDisc = new MusicFormat(203, "
  Violent Femmes", "American People"
  , MusicType.COMPACT_DISC, 8, 16);
record = new MusicFormat(204, "The
  Doors", "Morrison Hotel", MusicType
  .RECORD, 15, 30);
  }

  @Test
  public void hadFormatId(){
    assertEquals(204, record.
      getFormatId());
  }

  @Test
  public void hasArtistName(){
    assertEquals("Sonic Youth",
      cassette.getArtistName());
  }

  @Test
  public void hasAlbumName(){
    assertEquals("American People",
      compactDisc.getAlbumName());
  }

  @Test
  public void hasType(){
    assertEquals(MusicType.
      COMPACT_DISC, compactDisc.
      getType());
```

I&T I.T 3 & 4



pda_evidence_week3.rb

```
# I.T 3 & 4
# Demonstrate searching datat in a program.
# Function that searches data
# The result of the function running


rounds = [72,74,68,72,75, 71, 68, 71]

def golf_average_par(rounds)
  total = 0
  for round in rounds
    total = total + round
  end
  return total/rounds.length
end

result = golf_average_par(rounds).to_s + " is my average score per round of

puts result
puts "-----------------------------------------------------"
puts rounds.collect { |x| x.to_s + " is par for this round"}
puts "-----------------------------------------------------"
puts rounds.sort { |x,y| y <=> x}
puts "-----------------------------------------------------"
puts rounds.reverse
```

Evidence — craiglynagh@craigs-MacBook-Pro — ../PDA/Evidence — -zsh — 8...

```
74
72
→  Evidence git:(master) ✗ ruby pda_evidence_week3.rb
71 is my average score per round of golf!
-----------------------------------------------------
72 is par for this round
74 is par for this round
68 is par for this round
72 is par for this round
75 is par for this round
71 is par for this round
68 is par for this round
71 is par for this round
-----------------------------------------------------
75
74
72
72
71
71
68
68
-----------------------------------------------------
71
68
71
75
72
68
74
72
→  Evidence git:(master) ✗
```

ugh Toptal. No risk.

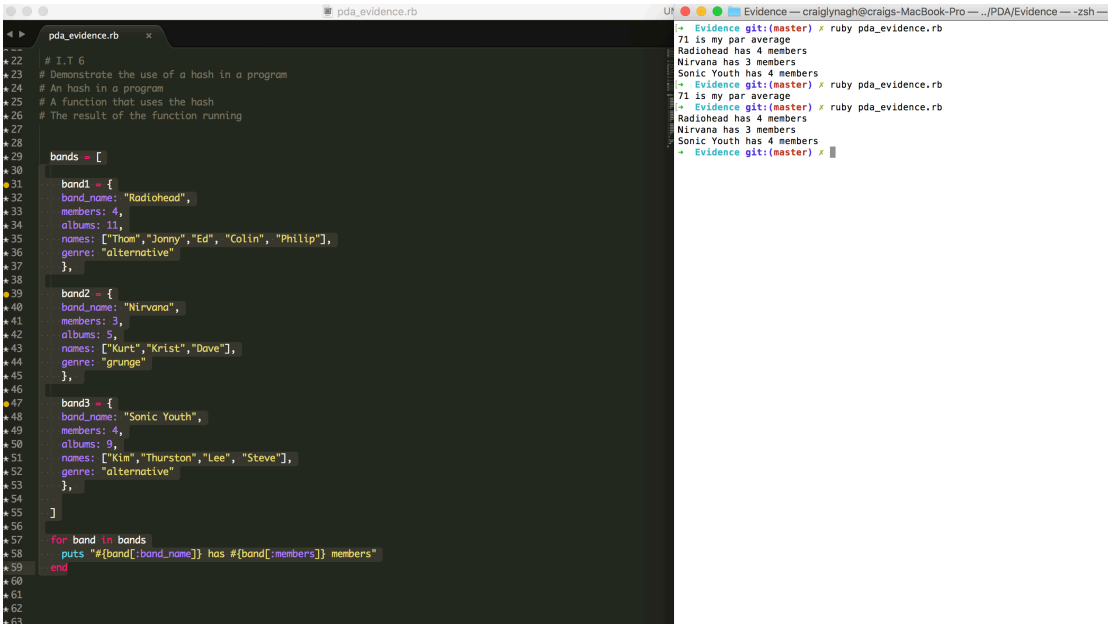
Craig Lynagh
PDA: Software Development

I&T: I.T 5

```ruby
# I.T 5
# Demonstrate the use of an array in a program
# An array in a program
# A function that uses the array
# The result of the function running


rounds = [72,74,68,72,75, 71, 68, 71]

def golf_average_par(rounds)
  total = 0
  for round in rounds
    total = total + round
  end
  return total/rounds.length
end

result = golf_average_par(rounds).to_s + " is my par average"

puts result
```

```
→  Evidence git:(master) ✗ ruby pda_evidence.rb
71 is my par average
Radiohead has 4 members
Nirvana has 3 members
Sonic Youth has 4 members
→  Evidence git:(master) ✗ ruby pda_evidence.rb
71 is my par average
→  Evidence git:(master) ✗
```

I&T: I.T 6

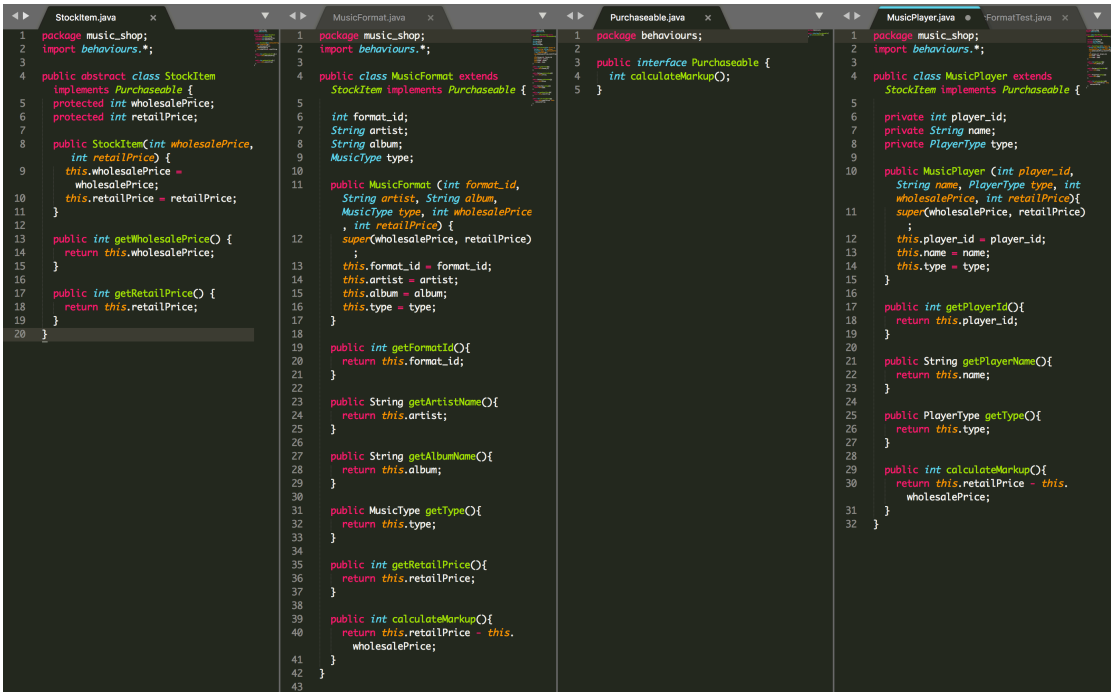


```ruby
# I.T 6
# Demonstrate the use of a hash in a program
# An hash in a program
# A function that uses the hash
# The result of the function running


bands = [

  band1 = {
  band_name: "Radiohead",
  members: 4,
  albums: 11,
  names: ["Thom","Jonny","Ed", "Colin", "Philip"],
  genre: "alternative"
  },

  band2 = {
  band_name: "Nirvana",
  members: 3,
  albums: 5,
  names: ["Kurt","Krist","Dave"],
  genre: "grunge"
  },

  band3 = {
  band_name: "Sonic Youth",
  members: 4,
  albums: 9,
  names: ["Kim","Thurston","Lee", "Steve"],
  genre: "alternative"
  },

]

for band in bands
  puts "#{band[:band_name]} has #{band[:members]} members"
end
```

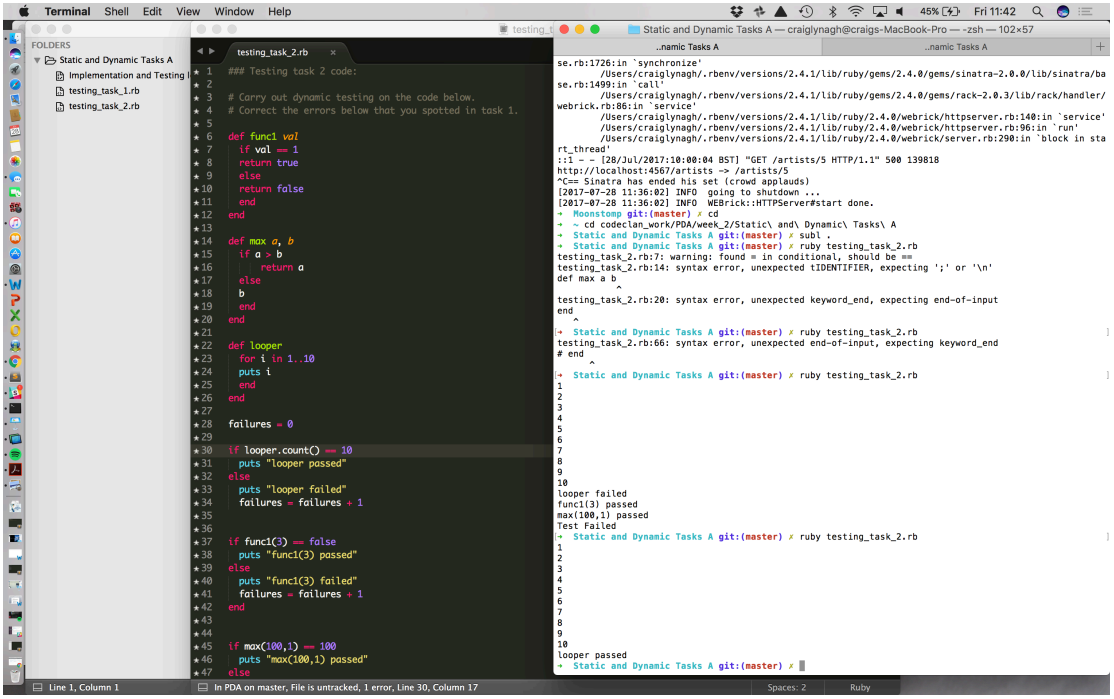
```
→  Evidence git:(master) ✗ ruby pda_evidence.rb
71 is my par average
Radiohead has 4 members
Nirvana has 3 members
Sonic Youth has 4 members
→  Evidence git:(master) ✗ ruby pda_evidence.rb
71 is my par average
→  Evidence git:(master) ✗ ruby pda_evidence.rb
Radiohead has 4 members
Nirvana has 3 members
Sonic Youth has 4 members
→  Evidence git:(master) ✗
```

Craig Lynagh
PDA: Software Development

I&T: I.T 7



I & T Static & Dynamic Testing Task A



Craig Lynagh
PDA: Software Development

Craig Lynagh
PDA: Software Development

Craig Lynagh
PDA: Software Development

Craig Lynagh
PDA: Software Development