W1004 Programming #4

Poker

Classes and their Constructors

- Imagine that you are the boss of a company
- Do you want to do a lot of work?
- No! So you hire workers
- These workers will be your other classes
- You create classes and instantiate classes (make objects of those classes) so you can divide the work amongst different parts of your program
- Classes should represent what they are and do what they should do (what should a Card class do)?

Classes and their Constructors

- A constructor is a recipe to create a new object of that class
  - Ex)Scanner in = new Scanner(System.in)
    - public Scanner(File source) {
      .....
      }
- This means that you can only create a Scanner object through the constructor
  - Cannot write
    - Scanner in = new Scanner(123);

Arrays

- Think of arrays as cabinets! You can put your shoes inside them (or whatever you want to)
  - int[] myNumbers = new int[10]
- Remember that indexes start at 0 not 1!
- You can access elements of array by
  - myNumbers[2]
    - accesses 2nd element of myNumbers
- Arrays cannot grow! (You have to copy all elements to a bigger array if you want to grow it)

ArrayList

- Arrays on steroids!
- You don't have to worry about the size of the array
- ArrayList also has lots of methods that you can use to help you out for this assignment
  - ArrayList<Card> cards = new ArrayList<Card>()
  - Collections.shuffle(cards);
  - cards.get(i);
  - cards.remove(i);
  - cards.set(i, newCard);
- Should I use arrays or arraylists?
  - It is up to you!

Let's Look an example program for BlackJack and dissect it

.....Look at the screen.....opening eclipse...

The Assignment

- What should each class do?
- Let's brainstorm!

compareTo()

- You can include a compareTo() method in a class
  - public int compareTo(Card other) {.....}
- If you do this then your class should implement comparable
  - public class Card implements Comparable<Card> {...}
- What does this mean?
  - This means that card implements the interface Comparable (more on interfaces later)
- What does this do for me?
  - Rather than getting info from each card and comparing them in your game class, you can have cards compare each other i.e.
    - int compare = cardOne.compareTo(cardTwo);
  - Depending on the int returned you can determine the relationship between cards (same value, same suit, etc.)

more on compareTo()

- If you implement compareTo() in a certain way then you can easily sort any list of cards
  - ArrayList<Card> cards = new ArrayList<Card>();
  - ……..
  - Collections.sort(cards);
- Note that this ONLY works if Card implements Comparable
- Your compareTo method should return a 1 if card1 is greater than card2, -1 if card1 is less than card2, and 0 if they are equal. i.e.
  - public int compareTo(Card other) {

```
        if(other.getValue() > this.value)
            return -1;
    ……..
    }
```

Questions?