COSC 111:
Computer Programming I

Dr. Bowen Hui

University of British Columbia
Okanagan

Today

- Review slides from week 2
- Review another example with classes and objects
- Review classes in A1

Discussion Forum on Connect

- Answers to questions

- TAs have posted helpful hints based on what they saw in the lab

Slides, Notes, Readings

- Course website's tentative lecture schedule updated as we go

- Slides and notes are posted
  - TAs will have access to what has been taught in class

- Read ***before*** class as part of your prep

From Week 2: Trivia Game Design

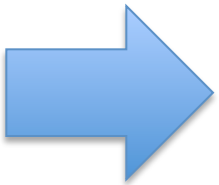
- Game
  - Main responsibility:
  - Needs to keep track of:
  - Have ability to do:
- Question
  - Main responsibility:
  - Needs to keep track of:
  - Have ability to do:
- Player
  - Main responsibility:
  - Needs to keep track of:
  - Have ability to do:

We said:
Once you have a design…

1. Translate each kind of object into a Java class
2. Translate group of things to keep track of into attributes
3. Translate each ability into a skeleton method
   - Detail the list of commands to issue in each method
   - Rule of thumb: each ability corresponds to one process (or routine, or method, or…)
     - Later, you will start to combine processes together
     - Later, you will start to decompose complicated processes into multiple, smaller processes

We said:
From Planning to Implementation

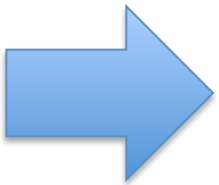
- Quiz game design
  - Keep track of
  - Abilities



```
class Game
{
    // attributes
    // methods
}
```

Class template

- Test design??



```
class TestQuiz
{
    Game quiz = new Game();
    quiz.startGame();
    // continue here
}
```

Simplified template

We had:
Classes for Game, Question, Player

```
class Game
{
    // attributes
    // methods
 }
class Question
{
    // attributes
    // methods
}
```

```
class Player
{
    // attributes
    // methods
 }
```

We had:
Methods

- Think of it as a routine
  - A set of repetitive steps grouped together
  - How to do something
- Used to define an object's ability
- Examples in your everyday life?
  - Carry conversation
  - Walking to a classroom
  - Cooking spaghetti
  - Doing homework

We introduced: Input-process-output (IPO)

- Model to show process interaction
- Input – information, resources needed
- Process – step-by-step commands
- Output – results of transforming I by P

We went through: Example IPOs for these

- Game ability
  - Display 10 questions in each game
- Question ability
  - Set correct answer
- Player ability
  - Maintain own high score across all previous games

We started: Translating to Java methods

- Template structure:

  returnType methodName($varType_1$ var_1, …, $varType_k$ var_k)

  {

  // commands go here

  }

Review: Example design

```
class Quiz
{
        // attributes …
        // constructor …
        // other methods here
        void startGame() { … }
        int getPlayerScore()
        {
                // access score in
                // person's attributes
                return -1;
        }
        int compHighs( int score1, int score2 )
        {
                // compare score1 with score2
                // return higher of the two
                return -1;
        }
        // continue next column
        boolean isGameOver()
        {
                // are all questions answered?
                // if so, return true
                return false;
        }
        String isBetter( Player p2 )
        {
                // compare my person's
                // high score to p2's highscore
                // return name of my person
                // if it has higher score
                // else return p2's name
                return "temp";
        }
}
```

Testing the classes

```
class TestQuiz
{
    Player bob = new Player( “robert” );
    // call its methods here

    Question q1 = new Question( “What is …”, “a. 1”, “b. 5”, “c. 10” );
    q1.setRealAnswer( “b” );
    // call its methods here

    Game quiz = new Game();
    quiz.startGame();
    // call its methods here
}
```

Calls the constructor method

Calls a method
pass in real values

We ended with this summary:

- Steps:
  - Convert class design to class skeleton
  - Convert each thing to keep track of into attributes
    - Declare them only
  - Build a constructor method
    - Initialize all the attributes in it
  - Convert each class ability into a method
  - Create a test class by creating object and calling its methods

```
class TestQuiz
{
        Quiz game = new Quiz();
        game.startGame();
}
```

Today: Another Game Example

```
class Question
{
  // attributes
  String questionWords;
  String mc1;
  String mc2;
  String mc3;
  int realAnswer;

  // methods
  Question( String q, String opt1, String op2, String op3 ) { … }

  String toString() { … }
}
```

Continue with Game Class

```
class Game
{
  // attributes
  int numCorrect;
  Question q1;
  Question q2;

  // methods
  Game() { … }

  void setAnswer1( int i ) { … }

  void setAnswer2( int i ) { … }

  void printQuestions() { … }
}
```

What do the Question objects look like?

- See handout:
  - Question-object-q1.txt shows the attributes have been set certain values that were used when q1 was initialized
  - Question-object-q2.txt shows different values that were used when q2 was initialized

Lastly, a TestGame class (simplified)

```
class TestGame
{
  // declare object here

  // call object's methods here
}
```

Lastly, a TestGame class (actual template)

```
class TestGame
{
  public static void main( String args[] )
  {
    // declare object here

    // call object's methods here
  }
}
```

TestGame class

```
class TestGame
{
  public static void main( String args[] )
  {
    // declare object here
    Game quiz = new Game();

    // call object's methods here
    quiz.setAnswer1( 2 );
    quiz.setAnswer2( 3 );
    quiz.printQuestions();
  }
}
```

Homework

- Study this latter Game example using the handouts we just reviewed (also available online)
  - Goal: Understand the detailed code with lots of comments
  - Use the comments as a guide to understand the program

Classes Provided in A1

- TakeOut.java
  - Review its responsibility, attributes, methods
  - Has most of the definition given
  - Some code missing
  - Use comments as a guide
- Customer.java and Order.java
  - Review their responsibilities, attributes, methods
  - Use comments as a guide
- TestTakeOut.java
  - Use comments as a guide
  - Some code missing