Ryan Gao and Shreyansh Thakral

03/11/2020
ICS4U0

Don't Fear OOP - Sci-Fi

Sci-Fi City

Every Sci-Fi City has certain characteristics to it: a name, a population, a technological advancement rating out of 10 that is relative to other sci-fi cities, a location, a time period, and a certain percent of the total population is Cyborgs. A typical Sci-Fi City has a population of 3,000,000 residents, a technological rating of 8.5, with 10% of the population being cyborgs and the year it is set in is 3200.

```
Sci-Fi City
        has a name
        has a population
        has a technological rating
        has a location
        has a year
        has a cyborg population percent

        A typical Sci-Fi City would have
                population = 3000000
                technological rating = 8.5
                cyborg population percent = 10%
                year = 3200
```

```
public class SciFiCity {
        String name;
        int population;
        double techRating;
        double cyborgPopPercent; //Percent value represented as a decimal
        String location;
        int year;

        public SciFiCity () {
                population = 3000000;
                techRating = 8.5;
                cyborgPopPercent = 0.10;
                year = 3200;
        }
}
```

Ryan Gao and Shreyansh Thakral

03/11/2020
ICS4U0

Human

All Humans have: a name, a number of arms and a number of legs, a gender, an energy level (in kJ), up to 5 minor technological modifications, a pay (per hour), and a bank balance. Humans typically start off with 2 arms, 2 legs, 0 minor modifications, and 100 kJ of energy. Humans can also perform actions, each of these actions will impact their characteristics appropriately. Humans can: eat (energy increases, money decreases), sleep (energy increases based on time slept), do work (money increases, energy decreases), have a minor modification done (money decreases, number of modifications increase).

Human
- has a name
- has a number of arms
- has a number of legs
- has a gender
- has a energy
- has a number of minor modifications
- has an hourly pay
- has an bank balance

A typical Human would have
- number of arms = 2
- number of legs = 2
- number of minor modifications = 0
- energy = 100

Eat, given a specific budget
- Energy increases by (4 + minor modifications /2) times budget
- Bank balance decreases by budget

Sleep, a specific number of hours
- Energy increases by (2 + minor modifications /2) times number of hours
- Energy gets capped at 300 if it is above this value

Work, for a specific number of hours
- Energy decreases by (4 - minor modifications/2) times hours
- Bank balance increases by pay times number of hours

Have a minor modification done

```
If 5 modifications have not been done, and bank balance >= 20,000 :
        minor modifications increases by 1
        bank balance decreases by 20,000
Otherwise if modifications are already 5:
        Print, "You have reached the human modification threshold, if you would like
        more modifications, please apply to be registered as a Cyborg."
Otherwise if bank balance < 20,000:
        Print, "You do not have enough funds to pay for this procedure."
```

```
public class Human
{
  String name;
  int numArms;
  int numLegs;
  String gender;
  int energy;
  int numMods;
  double hrPay;
  double bankBalance;

  public Human ()
  {
    numArms = 2;
    numLegs = 2;
    numMods = 0;
    energy = 100;
  }

  public void eat (double budget)
  {
    energy += Math.round ((4 + (double) numMods / 2) * budget);
    bankBalance -= budget;
  }

  public void sleep (int hrs)
  {
    energy += Math.round ((2 + (double) numMods / 2) * hrs);
    if (energy > 300)
```

```
        energy = 300;
    }

    public void work (int hrs)
    {
        energy -= Math.round ((4 - (double) numMods / 2) * hrs);
        bankBalance += hrPay * hrs;
    }

    public void haveMinorMod ()
    {
        if (numMods < 5 && bankBalance >= 20000)
        {
            numMods++;
            bankBalance -= 20000;
        }
        else if (numMods == 5)
            System.out.println("You have reached the human modification threshold, if you would like
more modifications, please apply to be registered as a Cyborg.");
        else if (bankBalance < 20000)
            System.out.println("You do not have enough funds to pay for this procedure.");
    }
}
```

Cyborg

A Cyborg is a Human that has had major technological modifications. Along with the characteristics of a Human, each Cyborg will have: a certain number of robotic arms, a certain number of robotic legs, at least 5 minor modifications, and a percent representing how much of their torso is robotic. Each Cyborg starts off with 0 robotic arms, 0 robotic legs, 2 legs, 2 arms and 5 minor modifications. Each robotic limb, and every 10% of robotic torso, counts towards 5 minor modifications. On top of the actions humans can perform, Cyborgs can: get an arm replaced (arms decreases, robotic arms increases, minor modifications increases, bank balance decreases) get a leg replaced (legs decreases, robotic legs increases, minor modifications increases, bank balance decreases), or get a torso modification (percent of robotic torso increases, minor modifications increases, bank balance decreases). Humans can be registered as Cyborgs if they meet the conditions of characteristics a Cyborg starts off with (0 robotic arms, 0 robotic legs, 2 legs, 2 arms and 5 minor modifications.)

Cyborg Extends the Idea of a Human, A Cyborg:
- has a number of robotic arms
- has a number of robotic legs
- has a percent of robotic torso

A Cyborg starts with
- number of arms = 2
- number of legs = 2
- number of robotic arms = 0
- number of robotic legs = 0
- number of minor modifications =5
- percent of robotic torso = 0.1
- energy = 100

Human transfers to a Cyborg, given the precondition that registering Human fits requirements
- number of arms = 2
- number of legs = 2
- number of robotic arms = 0
- number of robotic legs = 0
- number of minor modifications =5
- percent of robotic torso = 0.1
- name = name of registering human
- energy = energy of registering human
- gender = gender of registering human
- hourly pay = hourly pay of registering human
- bank balance = bank balance of registering human - 1,000 registration fees

Have a minor modification done
- If bank balance >= 20,000 :
  - minor modifications increases by 1
  - robotic torso percent increases by 0.02
  - bank balance decreases by 20,000
- Otherwise:
  - Print, “You do not have enough funds to pay for this procedure.”

Have a leg replaced:
- If bank balance >= 100,000 :
  - minor modifications increases by 5

```
                    number of robotic legs increases by 1
                    number of legs decreases by 1
                    bank balance decreases by 100,000
        Otherwise:
            Print, "You do not have enough funds to pay for this procedure."
        Have an arm replaced:
        If bank balance >= 100,000 :
            minor modifications increases by 5
            number of robotic arms increases by 1
            number of arms decreases by 1
            bank balance decreases by 100,000
        Otherwise:
            Print, "You do not have enough funds to pay for this procedure."

        Have a torso modification:
            If bank balance >= 100,000 :
                minor modifications increases by 5
                Robotic torso increases by 0.1
                bank balance decreases by 100,000
        Otherwise:
            Print, "You do not have enough funds to pay for this procedure."
```

```
public class Cyborg extends Human
{
  int botArms;
  int botLegs;
  double botTorso; // percent value represented as a decimal

  public Cyborg ()
  {
    numArms = 2;
    numLegs = 2;
    botArms = 0;
    botLegs = 0;
    numMods = 5;
    botTorso = 0.1;
    energy = 100;
  }
```

```
public Cyborg (Human registrant)
{
   numArms = 2;
   numLegs = 2;
   botArms = 0;
   botLegs = 0;
   numMods = 5;
   botTorso = 0.1;
   name = registrant.name;
   energy = registrant.energy;
   gender = registrant.gender;
   hrPay = registrant.hrPay;
   bankBalance = registrant.bankBalance - 1000;

}

public void haveMinorMod ()
{
   if (bankBalance >= 20000)
   {
      numMods++;
      botTorso += 0.02;
      bankBalance -= 20000;
   }
   else
      System.out.println ("You do not have enough funds to pay for this procedure.");
}

public void replaceLeg ()
{
   if (bankBalance >= 100000)
   {
      numMods += 5;
      botLegs++;
      numLegs--;
      bankBalance -= 100000;
   }
   else
```

```
        System.out.println ("You do not have enough funds to pay for this procedure.");
    }

    public void replaceArm ()
    {
      if (bankBalance >= 100000)
      {
        numMods += 5;
        botArms++;
        numArms--;
        bankBalance -= 100000;
      }
      else
        System.out.println ("You do not have enough funds to pay for this procedure.");
    }

    public void haveTorsoMod ()
    {
      if (bankBalance >= 100000)
      {
        numMods += 5;
        botTorso += 0.1;
        bankBalance -= 100000;
      }
      else
        System.out.println ("You do not have enough funds to pay for this procedure.");
    }
}
```

Vehicle

All vehicles have certain characteristics: model name, wheels, wings, total kilowatt-hour (kWh) battery capacity, current kWh out of the total capacity and passenger count. A standard vehicle has 4 wheels, 500 kWh total battery capacity and a passenger count of 6. All vehicles can travel (kWh decreases), charge (kWh increases, Human's money decreases), dock (passengers decrease to 0, kWh decreases at a minimal rate). When asked for the model name, the vehicle will display its model name.

```
Vehicle
        has a model name
        has wings
        has a certain number of wheels
        has a certain number of total kWh battery capacity
        has a certain number of current kWh
        has a certain number of passenger count

A standard vehicle will start with
        wheels = 4
        kWh battery capacity = 500
        passenger count = 6

travel
        current kWh decreases by 10

charge
        current kWh = total kWh battery capacity

dock
        passengers = 0
        current kWh decreases by 1

When asked for the model name
        tell them model name
```

```
public class Vehicle
{
   String model;
   int wingspan;
   int wheels;
   int kWhCapacity;
   int kWhCurrent;
   int passengers;

   public Vehicle ()
```

```
  {
    wheels = 4;
    kWhCapacity = 500;
    passengers = 6;
  }

  public void travel ()
  {
    kWhCurrent -= 5;
  }

  public void charge ()
  {
    kWhCurrent = kWhCapacity;
  }

  public void dock ()
  {
    passengers = 0;
    kWhCurrent--;
  }

  public String getModel ()
  {
    return model;
  }

}
```

Spaceship

A Spaceship is an advanced Vehicle for interplanetary travel. It retains the characteristics of a Vehicle along with: propulsion thrusters, total propellant capacity, current propellant out of the total capacity, oxygen level, cabin temperature (in °C) and supplies. In addition to the actions of a

Vehicle, a Spaceship can leave orbit (propellant decreases), travel to another planet (supplies decrease the further the distance travelled), replenish oxygen (oxygen level increases, supplies decrease, propellant increases to capacity) and restock (supplies increases).

Spaceship extends the idea of Vehicles

Spaceship
- has a certain number of thrusters
- has a certain number of total propellant capacity
- has a certain number of current propellant
- has a certain number of oxygen level
- has a certain number of cabin temperature
- has a certain number of supplies

A standard spaceship will start with
- thrusters = 4
- total propellant capacity = 10000
- cabin temperature = 20
- supplies = 100
- kWh battery capacity = 500
- wingspan = 10
- wheels = 0
- passengers = 50

leaveOrbit
- propellant decreases by 4000

travel
- supplies decreases 5
- current kWh decreases by 10

replenishOxygen
- oxygen increases by 1
- supplies decreases 1
- current propellant = total propellant capacity

restock
- supplies increased to 100

```
public class Spaceship extends Vehicle
{
  int thrusters;
  int propellantCapacity;
  int propellantCurrent;
  int oxygen;
  double temp;
  int supplies;

  public Spaceship ()
  {
    model = " ";
    thrusters = 4;
    propellantCapacity = 10000;
    supplies = 100;
    wingspan = 10;
    wheels = 0;
    kWhCapacity = 500;
    passengers = 50;
  }

  public void leaveOrbit ()
  {
    propellantCurrent -= 4000;
  }

  public void travel ()
  {
    supplies -= 5;
    kWhCurrent -= 10;
  }

  public void replenishOxygen ()
  {
    oxygen++;
    supplies--;
    propellantCurrent = propellantCapacity;
  }
```

```
  public void restock ()
  {
    supplies = 100;
  }
}
```

Time Travel Pod

A Time Travel Pod allows its user to travel forwards or backwards in time. All Time Travel Pods have certain characteristics: version number, travel direction, travel distance (in days), total plutonium cores, current plutonium cores and stability. A standard Time Travel Pod has 3 total plutonium cores and a full stability of 100. All Time Travel Pods can time travel (plutonium cores decreases, stability decreases), resupply (plutonium cores increases), receive service (stability increases). When asked for the version name, the Time Travel Pod will display its version name.

```
Time Travel Pod
        has a certain version number
        has a travel direction
        have a certain number of travel distance
        have a certain number of total plutonium cores
        have a certain number of current plutonium cores
        have a certain number of stability

A standard time travel pod will start with
        total plutonium cores = 3
        stability = 0
        passengers = 1
        wheels = 0
        wingspan = 0

timeTravel
        current plutonium cores decreases by 1
        stability decreases by 20

resupply
        current plutonium cores = total plutonium cores

receiveService
```

```
        stability increase to 100

When asked for the version name
        tell them version name
```

```
public class TimeTravelPod
{
  double version;
  String direction;
  int distance;
  int plutoniumCapacity;
  int plutoniumCurrent;
  int stability;

  public TimeTravelPod ()
  {
    plutoniumCapacity = 3;
    stability = 100;
  }

  public void timeTravel ()
  {
    plutoniumCurrent--;
    stability -= 20;
  }

  public void resupply ()
  {
    plutoniumCurrent = plutoniumCapacity;
  }

  public void receiveService ()
  {
    stability = 100;
  }

  public double getVersion ()
  {
```

```
    return version;
  }
}
```

Main Plot: The Rouge Cyborg

Here is the main plot of the Sci-Fi novel The Rouge Cyborg. It is set in the year 3200 in the City of Shryan. Shryan has a technological rating of 9.5, is located in California, and has all other characteristics in common with a typical Sci-Fi City.
There is a human named Viktor who is from Shryan, is a male, earns $25.00 per hour and has a bank balance of $15,000.00. Viktor has already had 4 minor body modifications, however, he would like to register as a Cyborg and have more. He works on a Spaceship that flies commercially to the moon. It must leave orbit, travel through space, dock, replenish oxygen and recharge. The Spaceship's model is Infinity XI. He works on the spaceship for a week after our story starts, wins a lottery, then registers as a cyborg and excessively modifies himself with his new abundant resources. Soon after he begins to regret his decisions and gets a time pod to travel back to try to reverse them. Will He be able to reverse his excessive decisions? Find out in part 2 of... The Rouge Cyborg

Here is the main plot at the Sci-Fi city. In the novel The Rouge Cyborg;

There is a SciFiCity called Shryan;
    Shryan is set in the year 3200;
    Shryan has a 9.5 technological rating;
    Shryan is located in California;

There is a Human called Viktor;
    Viktor's gender is male;
    Viktor's hourly pay is $25.00;
    Viktor's bank balance is $15000.00;

There is a Spaceship called Infinity;
    Infinity's model is Infinity X1;
    Infinity's current propellant is 8000;
    Infinity's oxygen level is at 100;
    Infinity's cabin temperature is at 20.0 °C;
    Infinity's supplies is 100;
    Infinity's wingspan is 50;
    Infinity's total propellant capacity is 1000;

```
There is a TimeTravelPod called Ion;
        Ion's version number is 2.08;
        Ion's travel direction is backward;
        Ion's travel distance is 365;
        Ion's current plutonium cores is 0;

For 7 days of a week:
        Display Infinity's info;
        Viktor eats;
        Viktor works;
        Infinity leaves orbit;
        Infinity travels through space;
        Viktor eats;
        Infinity travels through space;
        Infinity travels through space;
        Viktor eats;
        Infinity docks;
        Infinity replenishes oxygen;
        Infinity charges battery;
        Viktor sleeps;

Viktor's bank balance increases by $1,000,000,000;
Viktor registers as a cyborg;
Viktor replaces all limbs;
Viktor receives 8 torso mods;
Viktor resupplies the TimeTravelPod;
Viktor receives service for the TimeTravelPod;
Viktor travels back in time;
Shryan's year is decreased by 1;
```

```
public class Plot
{
   public static void main (String[] args)
   {
      SciFiCity shryan = new SciFiCity ();
      shryan.name = "Shryan";
```

```
shryan.techRating = 9.5;
shryan.location = "California";

Human viktor = new Human ();
viktor.name = "Viktor";
viktor.gender = "male";
viktor.hrPay = 25.0;
viktor.bankBalance = 15000.00;

Spaceship infinity = new Spaceship ();
infinity.model = "X1";
infinity.propellantCurrent = 8000;
infinity.oxygen = 100;
infinity.temp = 20.0;
infinity.supplies = 100;
infinity.wingspan = 50;
infinity.kWhCapacity = 1000;

TimeTravelPod ion = new TimeTravelPod ();
ion.version = 2.08;
ion.direction = "Backward";
ion.distance = 365;
ion.plutoniumCurrent = 0;

//A regular week
for (int i = 1 ; i < 8 ; i++)
{
   System.out.println ("Infinity XI: Log: Day " + i);
   System.out.println ("Preflight Check:");
   System.out.println ("O2:" + infinity.oxygen);
   infinity.charge ();
   System.out.println ("Battery:" + infinity.kWhCurrent);
   System.out.println ("Propellant:" + infinity.propellantCurrent);
```

```
        System.out.println ("Supplies:" + infinity.supplies);
        System.out.println ();
        infinity.passengers = 100;
        viktor.eat (10);
        viktor.work (9);
        infinity.leaveOrbit ();
        infinity.travel ();
        viktor.eat (15);
        infinity.travel ();
        infinity.travel ();
        viktor.eat (10);
        infinity.dock ();
        infinity.replenishOxygen ();
        infinity.restock ();
        System.out.println ("Viktor's Nightly Record:");
        System.out.println ("Energy: " + viktor.energy + " kJ");
        System.out.println ("Bank Balance: $" + viktor.bankBalance);
        System.out.println ();
        System.out.println ();
        viktor.sleep (8);
    }
    viktor.bankBalance += 100000000; // he wins a billion dollar lottery!
        System.out.println ("Viktor: I CAN'T BELIEVE IT I WON A BILLION DOLLARS I
WILL GET EVERY MODIFICATION POSSIBLE");
    viktor.haveMinorMod (); //He reaches human mod limit
    viktor.haveMinorMod (); // can't surpass limit without registering
    Cyborg vk = new Cyborg (viktor); // registers as cyborg
    vk.replaceLeg (); // replaces all limbs
    vk.replaceArm ();
    vk.replaceLeg ();
    vk.replaceArm ();
    for (int i = 0 ; i < 8 ; i++)
    {
```

```
        vk.haveTorsoMod ();  // has 8 torso mods, 90% of his torso is now robotic
      }
      System.out.println ("Viktor now has "+vk.botArms+" robotic arms, "+vk.botLegs+" robotic legs, and " + Math.round(vk.botTorso*100)+"% robotic torso");
      shryan.year++;  // a year passes
      System.out.println ("1 Year Later...");
      System.out.println ("Viktor: This is it, I can no longer live like this, I must go back and fix my mistakes, I must use the " + ion.getVersion () + " Ion Time Travel Pod to put an end to this");
      ion.resupply ();
      ion.receiveService ();
      ion.timeTravel ();
      shryan.year--;
   }
}
```