

CS-259

Data Structures with Java

Loops and Numerical Integration



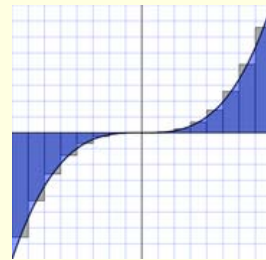
Instructor: **Joel Castellanos**

e-mail: joel@unm.edu

web: <http://cs.unm.edu/~joel/>

Course website:

<http://cs.unm.edu/~joel/cs259/>



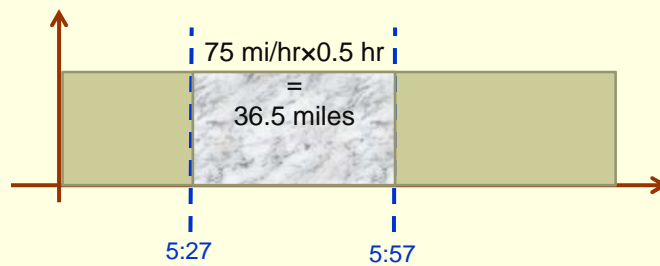
1

Upcoming Schedule

- *Today:*
 - Lab 10: Loops and Numerical Integration (due Thursday night).
 - *Homework 2*(due Friday): Carefully Read textbook Chapter 5: Inheritance, pages 171-185.
 - Only read “**C++ notes**” if you know C++.
- *Wednesday: Retest*
- *Friday: Quiz on Reading*

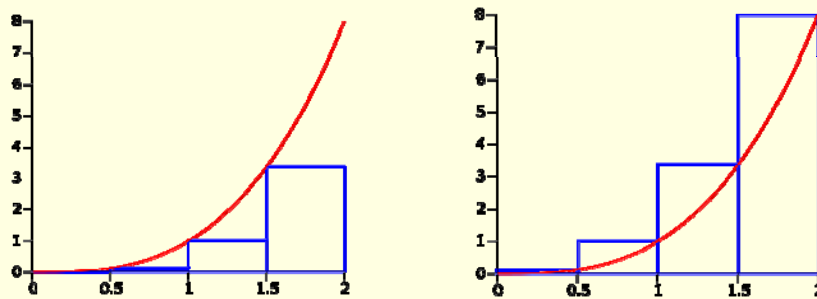
How Far does Alex Drive

- Alex is driving in cruise control on Interstate 40.
- His speed remains constant at 75 miles / hour from 5:00 PM through 7:00 PM.
- How much distance does Alex drive from 5:27 PM to 5:57 PM?



-3-

Upper and Lower Sum



-4-

Lab 10: How Far Did Alex Drive?

Given a method that returns the instantaneous velocity of Alex.

Write a method that uses numerical integration to find either the upper or lower bound of the distance Alex travels from one given time and another given time using a given number of rectangular steps.

-5-

Lab 10 API

```
public class MovingObject
{ double startTime = 0; //start time for the object

  public MovingObject(double t0)
  { startTime = t0;
  }

  // Returns velocity of this object in meters/second at the given number of
  // seconds, t, after this.startTime.
  public double getInstantaneousVelocity(double t)
  { //Unknown method you must call.
  }

  public double findDistanceUpperorLowerBound(double
  time1, double time2, int steps, boolean upper)
  { //This is the part you write
  }

  public static void main(String[] args)
  { //Used to test the class.
  }
}
```

-6-

Error Checking

```
throw IllegalArgumentException(message);

public double
  findDistanceUpperorLowerBound(
    double time1, double time2, int steps,
    boolean upper)
{ //This is the part you write
}
```

Input Values must be:

- steps >=1
- time1 <= time2
- time1 >= `this.startTime`

-7-

Example: `getInstantaneousVelocity`

```
public double getInstantaneousVelocity(double t)
{ return (t-startTime)*2.0;
  //return 5.0;
}
```

-8-

Algorithm

1. check for errors
2. $\text{deltaTime} = (\text{time2} - \text{time1}) / \text{steps}$
3. Loop $i=0$ to the number of steps
4. { speed1 =
 $\text{abs}(\text{getInstantaneousVelocity}(t1))$
5. speed2=
 $\text{abs}(\text{getInstantaneousVelocity}(t1 + \text{deltaTime}))$
6. speed = either max or min of speed1 and speed2
 depending on value of upper.
7. areaOfRectangle = speed * deltaTime
8. $t1 = t1 + \text{deltaTime}$
9. }
- 9- 10. return sum of all area of rectangles