CS415

INTRODUCTION TO COMPUTER SCIENCE

FALL 2017

3-OBJECT ORIENTED PROGRAMMING

SECTION 1.1-1.2

PREVIEW

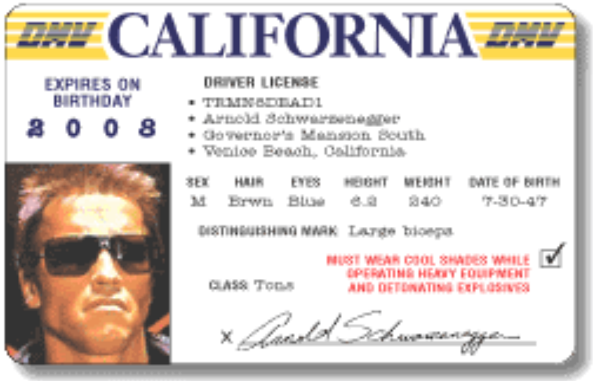
- Read Chapter 1
- What are models?
- Programs as models.
- Object Oriented Programming.
- Objects: Properties and Capabilities.

MODELS

- We use models to help us understand complex systems.
- Models leave out details and make it easier to understand the system.
- Most technical and scientific disciplines rely heavily on modeling.
- Modeling is at the heart of Computer Science.

> “Fundamentally, computer science is the science of abstraction---creating the right model for a problem and devising the appropriate mechanizable techniques to solve it.”
>
> A.Aho and J. Ullman

MODELS ARE EVERYWHERE



DMV CALIFORNIA DMV

EXPIRES ON BIRTHDAY
2 0 0 8

DRIVER LICENSE
- TRMN8DEAD1
- Arnold Schwarzenegger
- Governor's Mansion South
- Venice Beach, California

SEX	HAIR	EYES	HEIGHT	WEIGHT	DATE OF BIRTH
M	Brwn	Blue	6.2	240	7-30-47

DISTINGUISHING MARK Large biceps

MUST WEAR COOL SHADES WHILE OPERATING HEAVY EQUIPMENT AND DETONATING EXPLOSIVES ☑

CLASS Tons

X Arnold Schwarzenegger

Form 1040

Use this form

Form 1040 (20

Tax and Credits

Form 1040

Label (See instructions on page 16.)

Use the IRS label. Otherwis

LABEL

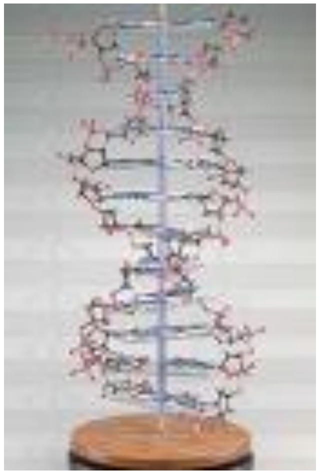
Department o

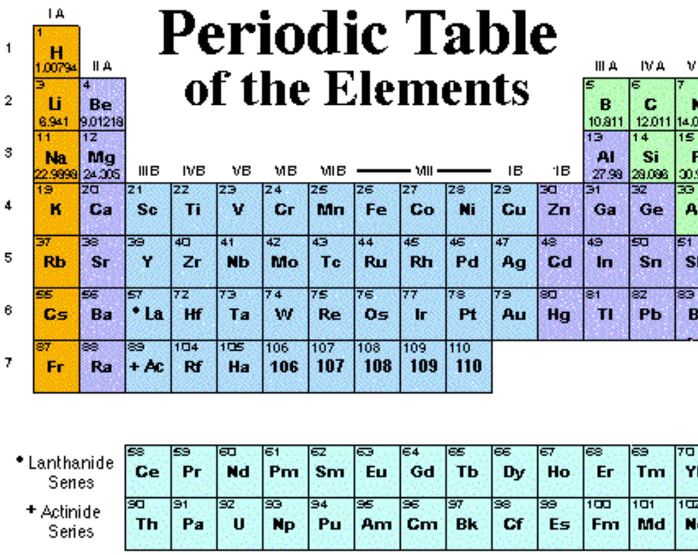
U.S. Indi

For the year Jan

Your first name

If a joint return, s

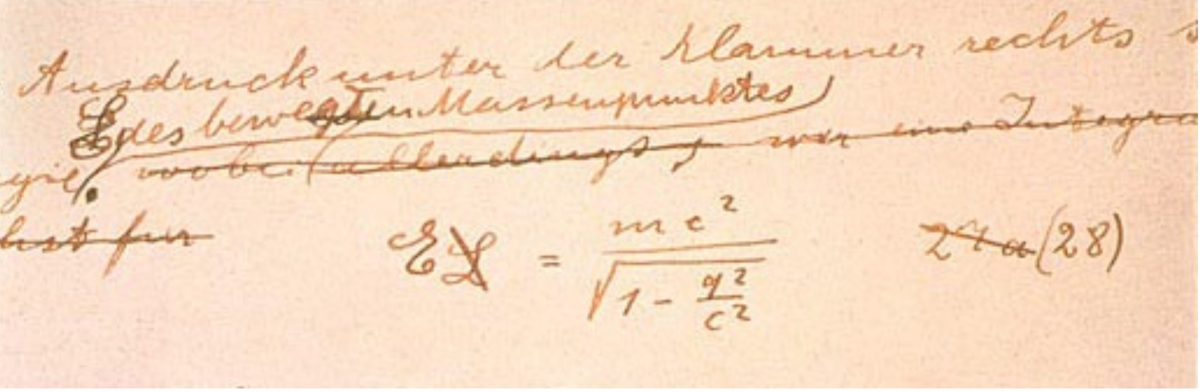




Periodic Table of the Elements

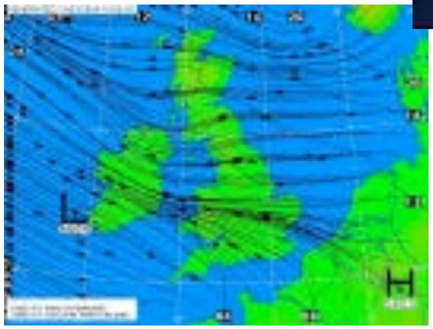
	I A	II A	III B	IV B	V B	VI B	VII B	VIII	VIII	VIII	I B	I B	III A	IV A	V
1	1 H 1.00794														
2	3 Li 6.941	4 Be 9.01218											5 B 10.811	6 C 12.011	7 N 14.0
3	11 Na 22.9898	12 Mg 24.305											13 Al 27.98	14 Si 28.086	15 P 30.9
4	19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 A
5	37 Rb	38 Sr	39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 S
6	55 Cs	56 Ba	57 • La	72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 B
7	87 Fr	88 Ra	89 + Ac	104 Rf	105 Ha	106 106	107 107	108 108	109 109	110 110					

• Lanthanide Series	58 Ce	59 Pr	60 Nd	61 Pm	62 Sm	63 Eu	64 Gd	65 Tb	66 Dy	67 Ho	68 Er	69 Tm	70 Y
+ Actinide Series	90 Th	91 Pa	92 U	93 Np	94 Pu	95 Am	96 Cm	97 Bk	98 Cf	99 Es	100 Fm	101 Md	102 N



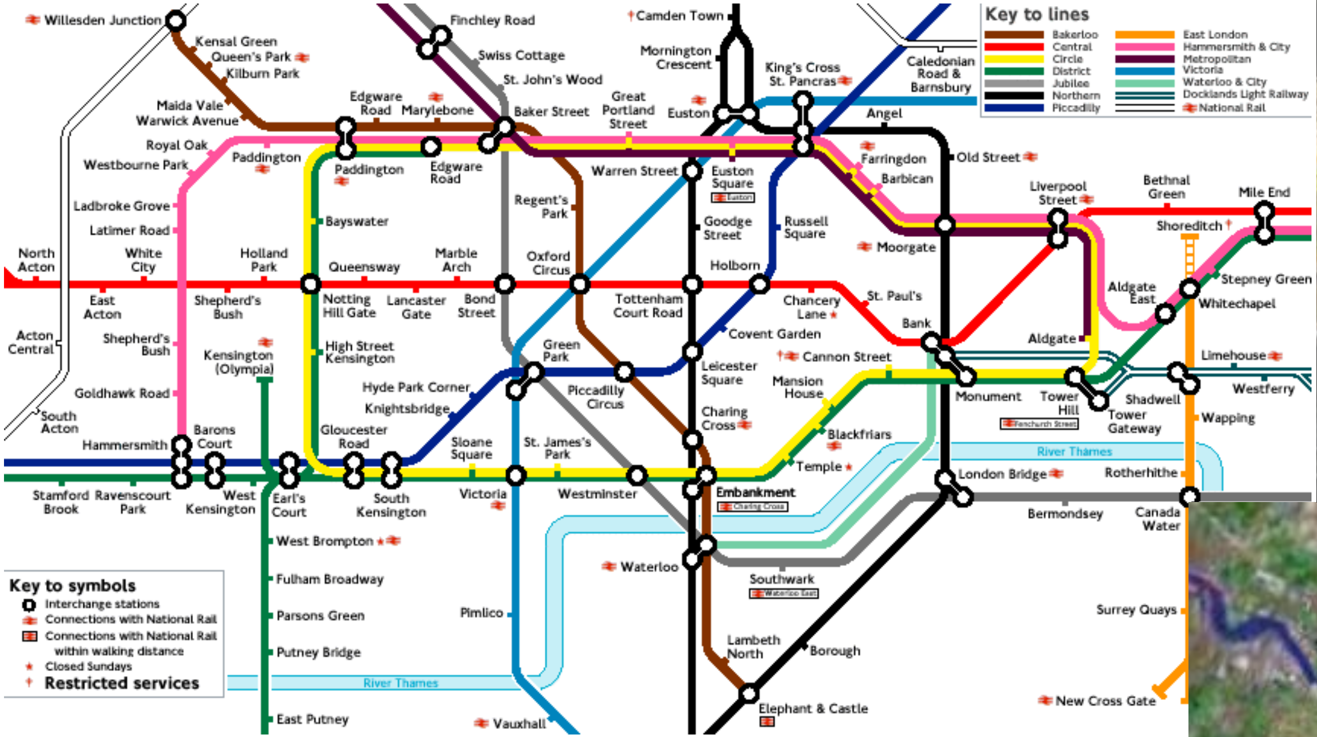
Ausdruck unter der Klammer rechts s
Edes bewegten Massenpunktes ~~war eine Integra~~
gie. ~~wobei (allerdings)~~
~~bist für~~

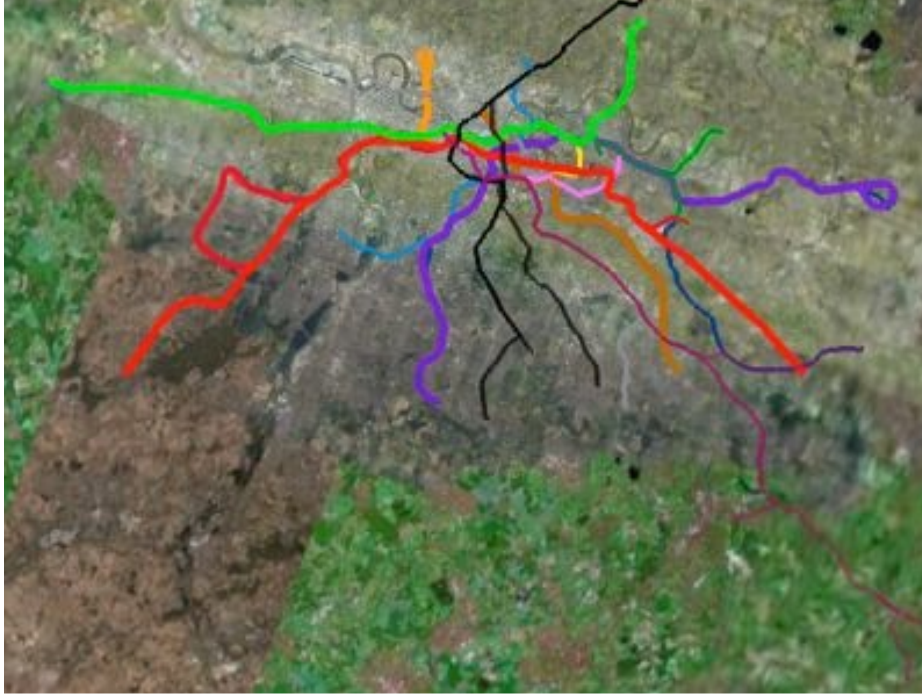
$$\mathcal{E}\cancel{L} = \frac{mc^2}{\sqrt{1-\frac{q^2}{c^2}}} \qquad \cancel{27a}\,(28)$$



MODELS

- A model takes a point of view.
  - which details are ignored?
  - which details are retained?





Maps are models:

What are the viewpoints of these two models of the London underground?

PROGRAMS AS MODELS

- We write a program to solve a problem within some system.

- The program models the system and then solves the problem within the model.

- Creating a good program begins with creating a good model.

MODELING A SYSTEM

- There are many different ways to create a model for a program.
- Programs today tend to be very large and need to be updated and improved often.
- The method that we will use is the best method we have (so far) for developing such programs.

OBJECT ORIENTED PROGRAMMING (OOP)

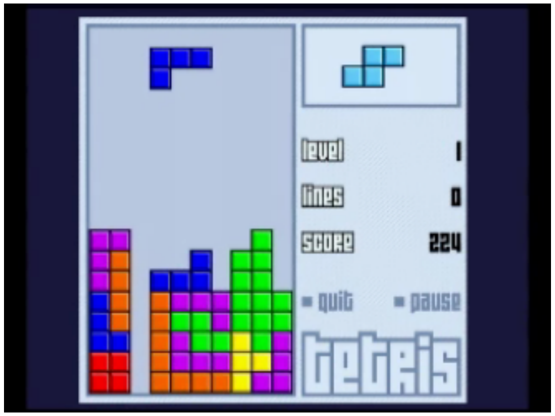
- In OOP we take the following view of the system being modeled:

  The system is a collection of objects which interact.

- The first steps in making a model is to decide on a point of view, analyze the system and pick out the important objects.

- Objects can model tangible things, conceptual things or even processes.

FIND THE OBJECTS...



- What are the objects in a Tetris game?

- A good approach is to describe the system and find the nouns

A random sequence of tetrominoes composed of four square blocks each, fall down the playing field. Each type of tetrominoes has a different color and different configuration. The objective of the game is to manipulate these tetrominoes, by moving each one sideways and rotating it by 90 degree units, with the aim of creating a horizontal line of blocks without gaps. When such a line is created, it disappears, and any block above the deleted line will fall. With every ten lines that are cleared, the game enters a new level. As the game progresses, each level causes the tetrominoes to fall faster, and the game ends when the stack of tetrominoes reaches the top of the playing field and no new tetrominoes are able to enter. The game can also end if the player is able to get all the way to level 15.

OBJECTS HAVE PROPERTIES

- Objects have properties that describe them.
- There are three types of properties:

1. Components: "parts" of the object
2. Attributes: "Adjectives" that describe the object
3. Peer Objects (Associations) :Things that the object "knows" about

PROPERTIES

A random sequence of tetrominoes, shapes composed of four square blocks each, fall down the playing field. Each type of tetromino has a different color and different configuration. The objective of the game is to manipulate these tetrominoes, by moving each one sideways and rotating it by 90 degree units, with the aim of creating a horizontal line of blocks without gaps. When such a line is created, it disappears, and any block above the deleted line will fall. With every ten lines that are cleared, the game enters a new level. As the game progresses, each level causes the tetrominoes to fall faster, and the game ends when the stack of tetrominoes reaches the top of the playing field and no new tetrominoes are able to enter. The game can also end if the player is able to get all the way to level 15.

OBJECT STATE

- The state of an object consists of the values of all the properties of the object.
- Some properties of an object might not be changeable, like the color of a J tetromino.
- Other attributes may change, like orientation of a tetromino

OBJECTS HAVE CAPABILITIES

- Objects can "do things", that is objects have capabilities.
- For instance, tetromino can rotate and fall.
- There are three types of object capabilities
  - Constructors: Create the object and Initialize the state
  - Accessors: Access the state and provide a value
  - Mutators: Change the state

CAPABILITIES

- We start with a description of the system and the nouns may become objects in our model.

- The verbs in the description may become capabilities of the objects.

- A capability (or behavior) of an object is something "it can do".

CAPABILITIES

A random sequence of tetrominoes, shapes composed of four square blocks each, fall down the playing field. Each type of tetromino has a different color and different configuration. The objective of the game is to manipulate these tetrominoes, by moving each one sideways and rotating it by 90 degree units, with the aim of creating a horizontal line of blocks without gaps. When such a line is created, it disappears, and any block above the deleted line will fall. With every ten lines that are cleared, the game enters a new level. As the game progresses, each level causes the tetrominoes to fall faster, and the game ends when the stack of tetrominoes reaches the top of the playing field and no new tetrominoes are able to enter. The game can also end if the player is able to get all the way to level 15.

TETROMINO OBJECTS

- Properties:
    - orientation (attribute)
    - color (attribute)
    - blocks (components)
    - board (peer)
- Capabilities:
    - initialize a new tetromino (Constructor)
    - rotate (mutator)
    - get board location (accessor)

CLASSES OF OBJECTS

- Note that a system may contain many objects of the same kind:

- The collection of all the objects of the same kind is called a class of objects

- Each object belongs to some class, we say that an object is an instance of its class

- The class of an object is also called its type

OBJECT CAPABILITIES

- All instances of the same class have the same properties
    - But, they may have different values for their properties
- All instances of the same class have the same capabilities:
    - But at any given time they may be doing different things

MANAGING COMPLEXITY

- Even after we eliminate details from the system we still may have a complex model.

- OOP offers us some techniques to help manage this complexity.

  - Encapsulation

  - Containment Hierarchies

  - Inheritance

MODEL DESIGN

- A system can have any number of models
  Some models will be better than others.
- If the system is divided into too few objects, each object will be too complicated and hard to understand.
- If a system is divided into too many objects, the objects will be simple but there will be too many interactions between them.

WHAT MAKES A GOOD SOFTWARE MODEL?

- Functionality
- Reliability (robustness)
- Usability
- Look and feel
- Efficiency
- Flexibility
- Maintainability

MODEL DESIGN

- The objects should hide their internal details from the other objects, this makes the overall system look simpler, and easier to change.
- The objects should be relatively independent of each other and interact is simple, predictable ways.

OBJECTS IN JAVA

- Objects have type:
  - In Java we will define a "Class" to specify an objects type.
- Objects have state (properties)
  - In a Java Class we will define "instance variables" to implement the objects state.
- Objects have behavior (capabilities.)
  - In a Java class we will define "methods" to implement the objects behavior

REVIEW

- Models give us a way to deal with a complex system.
- A program models a system.
- Object Oriented Programming organizes the system into interacting objects.
- Objects have state (properties) and behavior (capabilities.)

REVIEW

- Models give us a way to deal with a complex system.
- A program models a system.
- Object Oriented Programming organizes the system into interacting objects.
- Objects have properties and capabilities.

NEXT TIME

- Java: an Object Oriented Language.
    - Java will allow us to build an Object Oriented Programming model.
- In Java an object will be defined by a "Class"
- Object properties will be implemented by "instance variables"
- Object capabilities will be implemented by "methods".
- Read the remainder of Chapter 1.