CS106A

GObjects class

Name.method(...)

Object.setColor(Color.BLACK)

new GLabel(text, x, y)

label.setFont(family-style-size)

new GRect(x,y,width,height)

new GOval(x,y,width,height)

new GLine(xo,yo,x1,y1)

object.setFilled(fill) - fill can true or false, fills it black

object.setFillColor(color), change colors of the filling

object.setColor,for rectangle itself

getWidth(), provides the width of the graphics window

getHeight(), returns height

What matters is in the order that they are added.

Arithmetic expressions

% remainder, modulus operator 7%2= 1

Type cast- treat one type as another for one operation

Y= (double)x/2;  //y=1.5

Y=5.9;

X=(int)y; //y=5 , removes the decimal part without rounding

Expression short-hands

x += value; will add value to x

-= subtract from variable

\*= multiply

/= divide

Constants, should be CAPITALIZED

private static final double PI=3.14159;

boolean p=true;

p=(3<5);  then p=true

== equals to, to make a test for example a boolean

Short circuit evaluation, important for preventing errors in programming

Scope is the lifetime of a variable and is defined by the method it is created.

Cascading if

if(){} else if(){} else if(){}... When it finds the first true it moves on...

Switch statement, sugar because is not needed

Loop and a half

while (true) {

Int val = readInt(

If (val == SENTINEL) break;

Total += val;

}

Adding text

String str= "hello there";

println(str);

String str2= str + "!";

Methods opens the possibility to make abstractions and encapsulations.

A CD player is a method, because you provide arguments (CD) and depending on those you get different outputs (music).

A toast is another method.

double x= Math.pow(b,e);

receiver.message(arguments);

private void double feetToInches(double feet) {

return 12 \* feet;

}

Predicate method returns a Boolean

private boolean isOdd(int x) {

}

Method

You can call:

Parameters

Local variables

Constants and instance variables

Instance Variables

private int balance;

Local variables

Dclared inside method

Visible only in method

Alive in method

Local computation

Instance variables

Declared in class (not in method)

Visible in any method of the class

Lives as long as object lives

Store "state" of the object

Pseudo random

RandomNumber

import acm.util.\*;

private RandomGenerator rgen= RandomGenerator.getInstance();

rgen.setSeed(1) is good to test the program for bugs.

public class name extends superclass {

}

File name for class is "classname.java"

Whenever somebody discourages you from doing something, you should do it anyway...

public class MyCounter {

public MyCounter(int startValue){ // this is when a parameter is given

counter = startValue;

}

public MyCounter() { // this is when a parameter is not given

counter = 1;

}

private int counter;

}

When you pass objects, you pass the original, this is what happens with class

With int, double, boolean you pass by copy value

With classes you pass by reference

Constructor, special kind of method, a factory.

The class has the same name as the method

Constructor does not specify return type

Responsible for initializing object

It is called when object is created (new called)

When a parameter has the same value as an instance variable. The parameter takes precedence over the instance

variable.

This solves it.

this . counter = counter;  this will focus on the instance variable.

Just make the name different for making it easy.

private static int counter; makes the a variable that all refer to.

String line = readLine();

Javadoc comments starts with: /\*\*  and ends \*/

Getters and setters method that get info and others that can change or set information in variables

For breakout! We could create a class for the ball

Layering is called the stacking order z-ordering

GCanvas

Used to represent the background canvas of collage

getElementAt ( x , y ) ; it returns a GObject, if there's nothing there then it will return null. Very useful for Breakout!

Interface is a set of method (headers)

Is a set of common functionality

GFillable is an interface that can be used in GRect, GOval, GArc and GPolygon

Event driven programs are those that are triggered by the user

GArc class

For the arc we specify a starting point and then a sweep of the arc. And before that we specify the angles, the diameter is established of the oval.

GArc a2 = new GArc( d, d, 0, 90) ;

GImage name of the file and x and y

GPolygon

Has a reference point

First create an empty gpolygon

Then add vertices one at a time using addVertex (x,y) and x and y are relative to the reference point of polygon

The reference point is the base for locating the polygon.

GCompound

You add stuff to the compound, not the canvas.

Then you have to add it to the canvas.

Enumeration

It's a mapping from concepts to numbers.

Characters are noted with single quotes

ASCII guarantees that 0 to 9, a to z and A to Z are sequential

Char - character, must use single quote

for ( char ch = 'A' ; ch <='Z' ; ch++)

To make lower case

ch = (ch - 'A') + 'a' ;

string s = ...

char ch= s . charaAt (0) ;

char is a primitive type, whereas string is a class / objects

But Character is a class full of methods to work on chars

Strings are immutable, in order to change them they have to be reassigned

Str = str . toUpperCase () ; IT HAS TO BE ASSIGNED TO A VARIABLE BECAUSE THEY ARE IMMUTABLE

s . substring (2 , 6) and it gives you the substring without including the last character in position 6 in this case. This value must be assigned to a variable.

As strings are objects and not primitives variables they must checked with a method as if ( s1 . equals (s2)) {...} it is case sensitive

How to have a double quote in a string you type it like : "\" " with a backward slash.

Tokenizing - breaking stuff into its parts

 StringTokenizer nextToken gives you each word of the string

xkcd blog

Encryption

Caesar cipher - rotate alphabet by n letters

Bit: a 0 or 1

Byte: 8 bits

Word: java 4 bytes

Stack: local variables, parameters are allocated to the stack. They are automatically reclaimed by the computer.

Heap: is where objects live in the memory. Keeps track of objects by their address. Reclaimed when object is done.

Heap and stack collision.

Pointee: is the actual object

Pointer: has the address of the pointee.

When an object is not declared it is null and the it will return an error if a method is called upon them.

When you pass a primitive, you pass a copy of the value, whereas when you pass an object, you pass the address for its location, therefore you change the actual object.

S1= "hi"

S2="hi"

S1 == S2 ? NO!

For strings you have to use if (S1 . equals (S2)) then...

Files

Text files,

Reading a file 3 steps involved:

1- open (BufferedReader) import java.io. \* ;

2- read

3- close

Potentially dangerous code you put it in a try,  and catch

Throw new ErrorException (ex);

Writing a file

1- open

2- write

3- close

Array list

List of information to keep track of. Caveats:

Ordered

Homogeneous

Import java.util.\*;

A list dynamically changes its size

It has a functionality

Template: here is how something work,

Java 1.5 added templates

Template <T> the t is for type.

ArrayList<String> strList =  new ArrayList <String> () ;

Array lists can only work with class, i. E. objects. No primitives, no ints.

In order to store primitives in an array

Class (object)

Integer

Double

Boolean

Character

Integer x = new Integer (5) ;

int y = x . intValue () ;

But it can be done automatically the normal way. Like

Integer x = 5 ;

int y = x ;

x = x + 1 ;

These wrapper classes are immutable.

Boolean add and remove, true means that the list was modified so add is always true. Whereas remove can be false when it doesn't find the element specified to remove so it doesn't remove it.

Set overwrites on top of the previous value

ArrayList <GLabel> labels = new ArrayList <GLabel> () ;