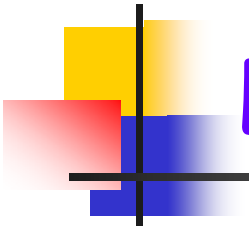
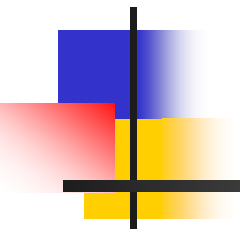
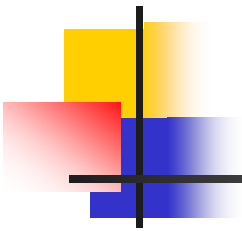


Lecture3

- Functions
- Strings
- Arrays
  - 2D and multi-dimensional arrays
- OOP concepts of Java
  - class and object



Functions



Java functions

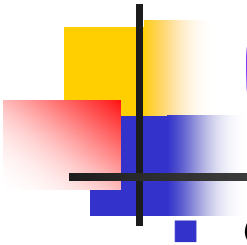
- A group of statements
  - To perform a task
  - Possibly return a value
- Unlike C, functions are part of a class in Java

- Syntax

```
<return_type> fn_name (arguments)
    {
        // function body
    }
```

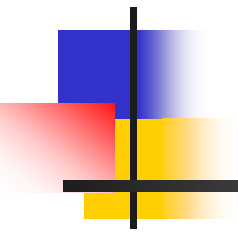
Java functions … example

```
import java.io.*;

public class anyClass                          // Define a class first
{
   int square (int x)                          // function to compute square
     {
        return (x * x);
     }

   public static void main(String args[ ])  //Starting point of the program
     {
        int i = 5;
        anyClass ac = new anyClass();       // Create an object first
        int i_sq = ac.square (5);           // Call its function
        System.out.println ("Square of 5 is: " + i_sq);
     }
}
```

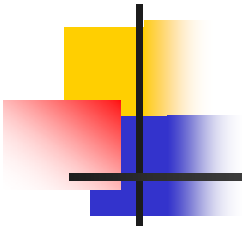


Useful Java functions

- clone ( )
  - Create a copy of an existing object
- equals ( )
  - Checks if two objects are the same
  - This is not quite the same as == operator
- finalize ( )
  - Called to clean up object's resources.
- getClass ( )
  - Returns a class object
- hashCode ( )
  - Returns object's memory address in hexadecimal
- toString ( )
  - Returns a string representation of the object.

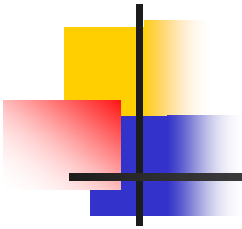


OOP – Class and objects



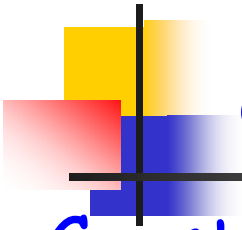
Class and objects

- class – the basic unit of OOP in Java
- A class typically corresponds to some meaningful entity.
- class has both data and methods.
- Attributes and methods are members of a class
- An instance of a class is an object.
- A class uses methods to interact with other classes/functions.



Class and objects … contd.

- Classes may have both
  - Data attributes
    - Can be of basic or user defined data types.
    - Need to be initialized – typically done in a constructor
  - Methods, Functions
    - Functions that are part of classes
    - Typically interfaces to interact with other classes and functions.
    - Provide APIs to external world to access and manipulate data attributes.



Constructor and destructor … contd.

Constructor

- A function with the same name as the class
- Called when an object is created
- A class can have more than one constructor

Destructor

- There is NO destructor in Java, equivalent to C++ destructor
- In C++
  - Destructor: A function with the name ~classname( )
  - Called when the object goes out of scope, or deleted.
- In Java, closest equivalent is finalize( ) function
  - Used to clean up system resources
    - E.g. close open files, open sockets
    - Clear screen for GUI/graphics objects.
  - Called by system garbage collectors and other resource cleanup functions.

A simple "account" example

```
public class Account
{
  private  int user_SSN;                                  // attribute (data)
  private  int accountNumber;                             // attribute (data)
  public   void withdrawMoney (int amount) { .. } ;       // method
  public   void depositMoney (int amount)   { .. };       // method
  public   void computeInterest( )          { .. };       // method

  public Account() { }                                    // Constructor
  public static void main (String args[ ])                // main function
    {
      Account a = new Account( );                         // Create a new object
      System.out.println ("In account main");
    }
};
```

Account example … contd.

```
import java.io.*;

public class Account
{
  private  int user_SSN;              // attribute (data)
  private  int accountNumber;         // attribute (data)
  public   myClass m = null;

  // Account constructor
  public Account( )
    {
       user_SSN = -1; accountNumber = -1; // default values
    }

  // Account finalize function
  protected void finalize( )
    {
       System.out.println ("In Account finalize function");
    }

  // Account main function
  public static void main (String args[ ])
    {
       Account a = new Account( );
       System.runFinalizersOnExit(true); // deprecated
    }
};
```

Class definition

Constructor

finalize function

main function

Calls finalize functions