Learning Outcomes

• A basic understanding of object-oriented programming concepts.
• An ability to create objects and call methods on objects.
• Familiarity with the graphics library included with the textbook.

• Adapted from slides by Sue Evans et al.
Procedural Programming

• The classic style of computer programming is called **procedural programming**.
• The workings of a program are defined in terms of sequences of operations, defined in a hierarchy
  – functions, which perform a sequence of operations, interspersed with calls to nested functions, which in turn call other functions
• Data is a *separate* group of entities, which are passive, and acted upon by procedures

Object-Oriented Programming

• More recently, development of an object-oriented programming (OOP) paradigm
  – Data-centric model of the world
  – Data and methods are combined into *objects*.
  – Objects *know stuff* (have state) and can *do things*.
  – Objects *interact* by sending each other messages.
  – A message is a request for an object to perform one of its actions.
  – Typically, sending a message to an object invokes a *method*—a data-specific procedure

Procedural vs. OOP

• Take a bank account program
• Procedural model: you write a function that looks up the account record, subtracts the deposit to the balance, and writes it back into the record
• OOP: the main method sends a message to the BankAccount object to transact a deposit
• Underneath it all, still procedural code
A Toy OOP Example

A bank account class:
• Class: BankAccount
• Creation: BankAccount(customerName, startingBalance)
• State data:
  – accountNumber
  – name
  – balance
• Methods: makeDeposit, makeWithdrawal, getBalance

A Toy OOP Example

• To create a new object, we call a special creation method, called a constructor:
  – Form: <className>(<param1>, <param2>, …)
  – E.g.:
    acct1 = BankAccount("John Doe", 100)
    acct2 = BankAccount("Jane Smith", 200)

A Toy OOP Example

• To have the object do things, we invoke a method, by using the dot notation:
  – Form: <object>.<methodName>(<param1>, …)
  – E.g.:
    >>> acct1.getBalance()
    100
    >>> acct1.makeDeposit(500)
    >>> acct1.getBalance()
    600
    >>>
OOP and Graphics

• We are going to explore objects in the motivating context of simple graphics programming.

• Notational conventions in the following slides:
  – <class-name> refers to the class that we want to create a new instance of. For example, GraphWin is a class name we will use often.
  – <param1>, ... refer to parameters that can be passed to the constructors and methods. The number and type of the parameters differs based on the class and method

[Please continue with the online slideshow available HERE]