Common Pitfalls and Errors

Hui Chen

Department of Computer & Information Science

Brooklyn College

Objectives

- To avoid common errors and pitfalls in elementary programming (§2.18).

Outline

- Discussed
  - Problem → Algorithm → Implementation
  - Design a program with input and output
  - Numeric data types and operators
  - Augmented assignment
  - Type casting
  - System time and conversion
- This lesson covers an example
  - Pitfalls and errors

Problem. Compute Loan Payment

- This program lets the user enter the interest rate, number of years, and loan amount, and computes monthly payment and total payment

$$monthlyPayment = \frac{loanAmount \times monthlyInterestRate}{1 - \frac{1}{(1 + monthlyInterestRate)^{numberOfYears \times 12}}}$$

```
import java.util.Scanner;

public class ComputeLoan {
  public static void main(String[] args) {
    // Create a Scanner
    Scanner input = new Scanner(System.in);

    // Enter yearly interest rate
    System.out.print("Enter yearly interest rate, for example 8.25: ");
    double annualInterestRate = input.nextDouble();

    // Obtain monthly interest rate
    double monthlyInterestRate = annualInterestRate / 1200;

    // Enter number of years
    System.out.print(
      "Enter number of years as an integer, for example 5: ");
    int numberOfYears = input.nextInt();

    // Enter loan amount
    System.out.print("Enter loan amount, for example 120000.95: ");
    double loanAmount = input.nextDouble();

    // Calculate payment
    double monthlyPayment = loanAmount * monthlyInterestRate / (1
      - 1 / Math.pow(1 + monthlyInterestRate, numberOfYears * 12));
    double totalPayment = monthlyPayment * numberOfYears * 12;

    // Display results
    System.out.println("The monthly payment is $" +
      (int)(monthlyPayment * 100) / 100.0);
    System.out.println("The total payment is $" +
      (int)(totalPayment * 100) / 100.0);
  }
}
```

Is there any problems in the implementation?

Problem. Monetary Units

- This program lets the user enter the amount in decimal representing dollars and cents and output a report listing the monetary equivalent in single dollars, quarters, dimes, nickels, and pennies. Your program should report maximum number of dollars, then the maximum number of quarters, and so on, in this order.
- But, …

Common Errors and Pitfalls

- Common Error 1: Undeclared/Uninitialized Variables and Unused Variables
- Common Error 2: Integer Overflow
- Common Error 3: Round-off Errors
- Common Error 4: Unintended Integer Division
- Common Error 5: Redundant Input Objects

- Common Pitfall 1: Redundant Input Objects

Questions?

- Let’s avoid common pitfalls and errors.

Common Error 1

- Undeclared/Uninitialized Variables and Unused Variables

double interestRate = **0.05**;

double interest = interestrate * **45**;

Common Error 2

- Integer Overflow

```
int value = 2147483647 + 1;

// value will actually be -2147483648
```

Common Error 3

- Round-off Errors

- Have you seen these?

```
System.out.println(1.0 - 0.1 - 0.1 - 0.1 - 0.1 - 0.1);

System.out.println(1.0 - 0.9);
```

Common Error 4

- Unintended Integer Division

```
int number1 = 1;
int number2 = 2;
double average = (number1 + number2) / 2;
System.out.println(average);
```

(a)

```
int number1 = 1;
int number2 = 2;
double average = (number1 + number2) / 2.0;
System.out.println(average);
```

(b)

Common Pitfall 1

- Redundant Input Objects

```
Scanner input = new Scanner(System.in);

System.out.print("Enter an integer: ");

int v1 = input.nextInt();

 Scanner input1 = new Scanner(System.in);

System.out.print("Enter a double value: ");

double v2 = input1.nextDouble();
```

Common Pitfall 2

- Violating programming convention
  - Identifiers?
  - Indentations?

Questions?