More Basic Python Programs, Assignments and Loops

Rose-Hulman Institute of Technology

Computer Science and Software Engineering

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

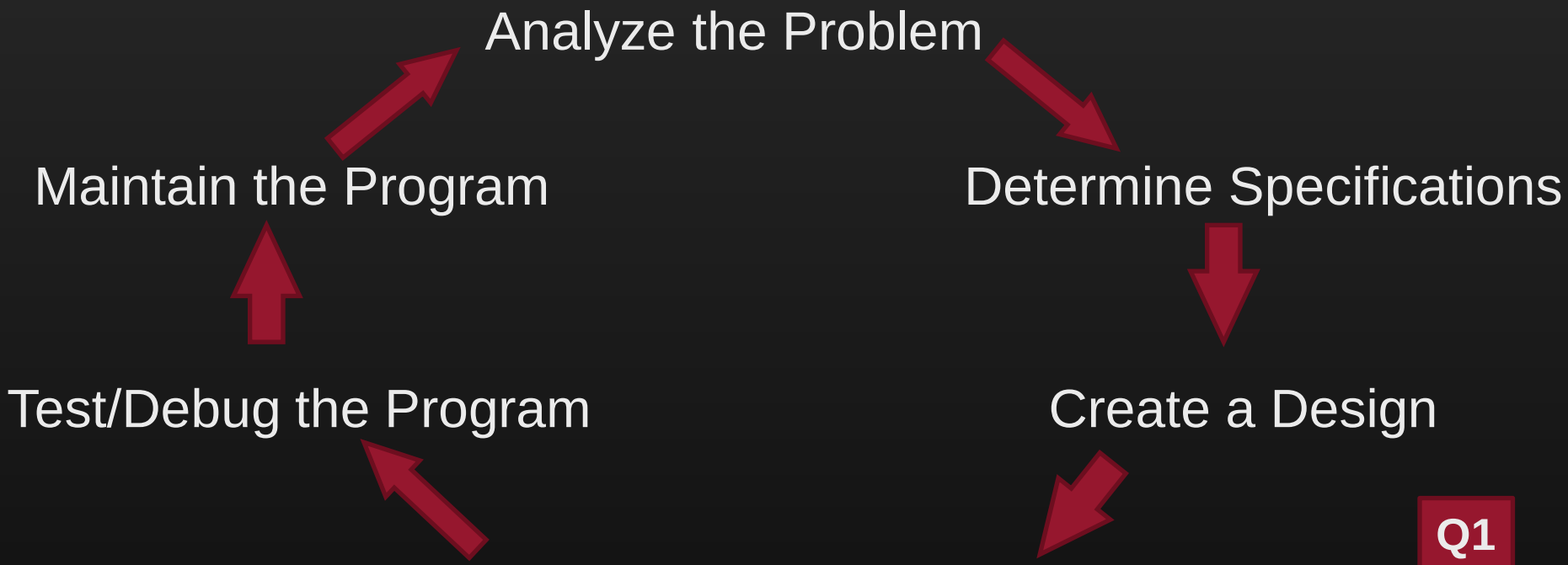
Outline

- Software development process
- Names, Expressions, output statements
- Basic number types: **int** and **float**
- Variables and assignments
- Definite loops

Check out 03-AssignmentsAndLoops

- Go to SVN Repository view at bottom of the workbench
  - Missing? Add it back:
    Window→Show View→Other→SVN → SVN Repositories
- Browse SVN Repository view for **03-AssignmentsAndLoops**
- Right-click it and choose **Checkout**
- Accept defaults in the dialog
- Expand the **03-AssignmentsAndLoops** project that appears in **Package Explorer** (on the left-hand-side)

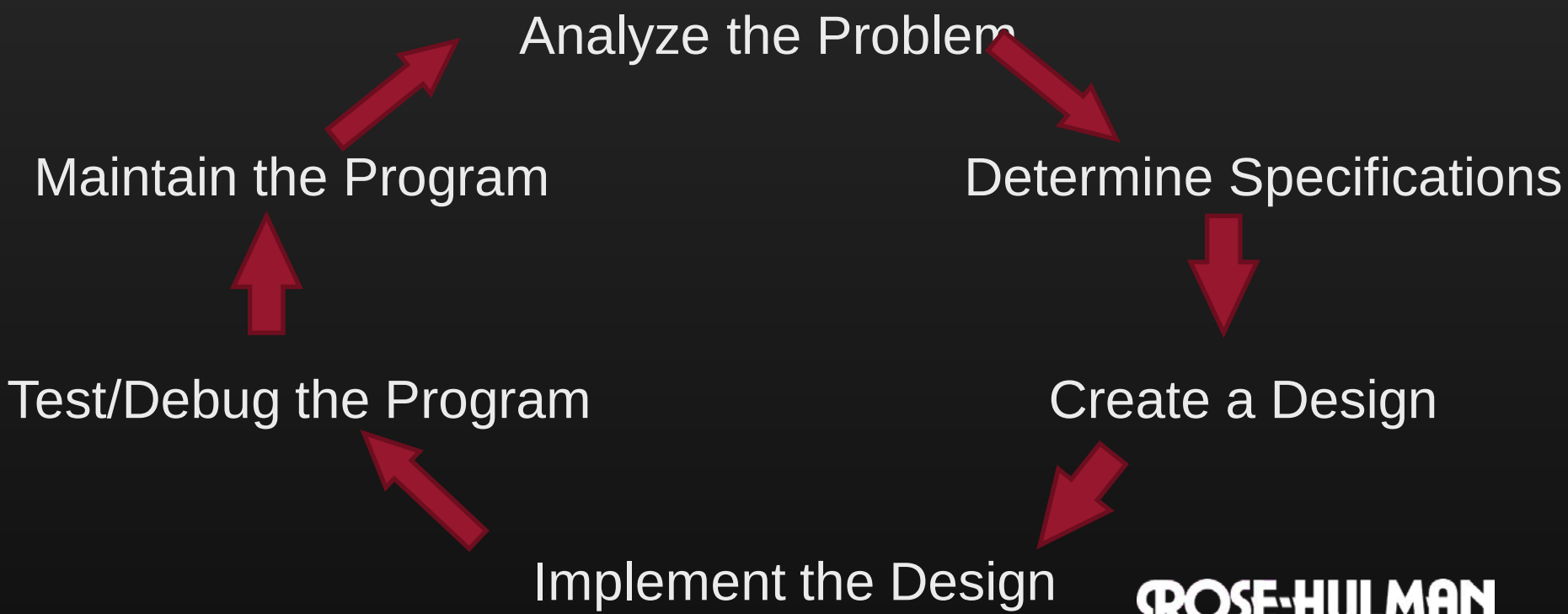
The Software Development Process



Susan's European Adventure

ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

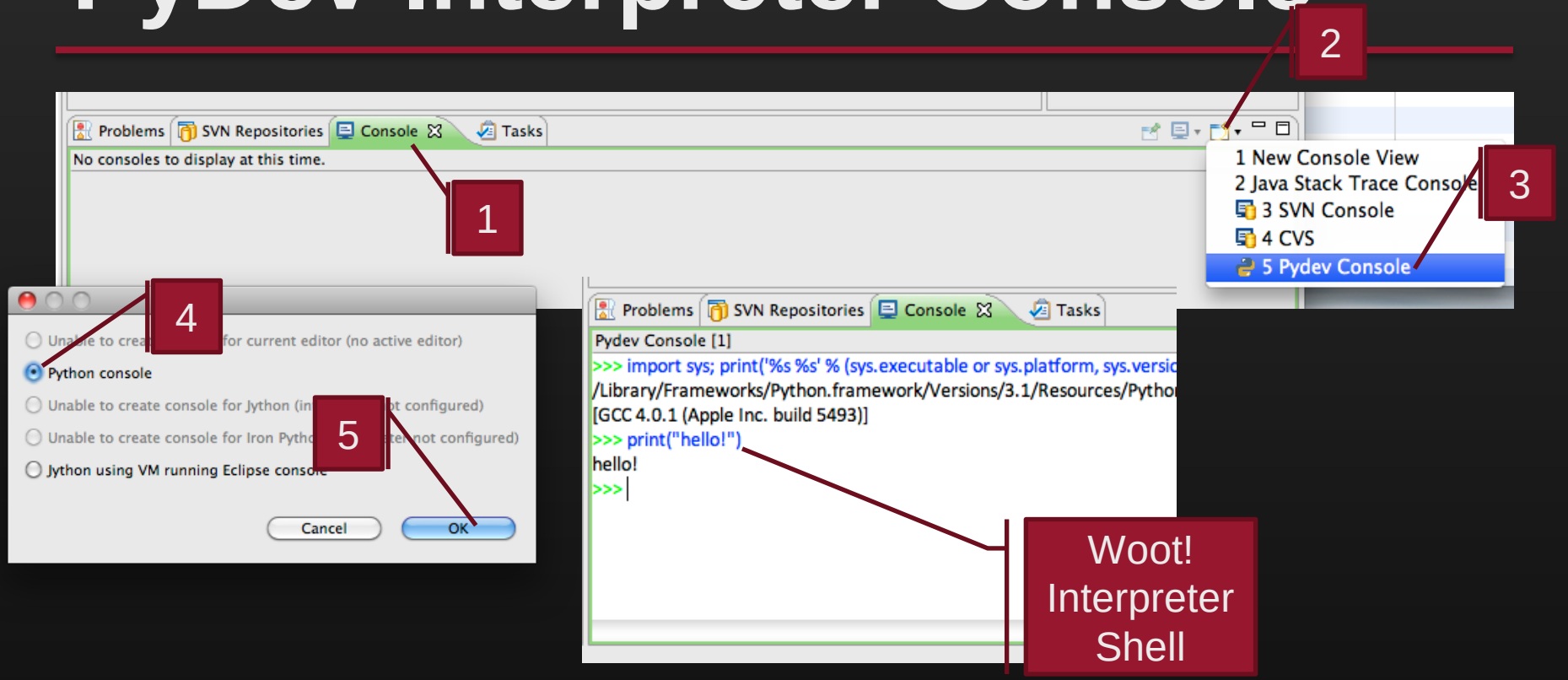
The Software Development Process



More input-compute-output practice

- Examine the **temperature.py** module in your Eclipse project
- Do the TODOs in it

PyDev Interpreter Console



ROSE-HULMAN
INSTITUTE OF TECHNOLOGY

Some numeric operations

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Exponentiation
%	Remainder
//	Integer division

Function	Operation
abs(x)	Absolute value of x
round(x, y)	Round x to y decimal places
int(x)	Convert x to the **int** data type
float(x)	Convert x to the **float** data type

Identifiers: Names in Programs

- Uses of identifiers so far…
  - Modules
  - Functions
  - Variables
- Rules for identifiers in Python
  - Start with a letter or _ (the “underscore character”)
  - Followed by any sequence of letters, numbers, or _
- Case matters! **spam ≠ Spam ≠ sPam ≠ SPAM**
- Choose descriptive names!

Q2a

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Reserved Words

- Built-in names, can't be used as identifiers
- Python reserved words:

and	del	**for**	is	raise
assert	elif	from	lambda	return
break	else	global	not	try
class	except	if	or	while
continue	exec	**import**	pass	with
def	finally	**in**	**print**	yield

Q2b

Don’t redefine function names!

- Examples that trip me up:
  - *len* – used to find the length of a list
  - *max*
  - *min*

Expressions

- Fragments of code that produce or calculate new data values
- Examples
  - *Literals*: indicate a specific value: **2**, **3.5**, **"hello"**
  - *Identifiers*: evaluate to their assigned value: **x**, **width**
  - *Compound expressions* contain other expressions:
    - E.g., **3 + 4 * 2**, **x < 5**
- Can use parentheses to group: **(3 + 4) * 2**

Q3,4

Recall…

- Programming languages have precise rules for:
  - Syntax (form)
  - Semantics (meaning)
- Computer scientists use *meta-languages* to describe these rules
- Example…

Output Statements

- Syntax:
  - **print()**
  - **print(<expr>)**
  - **print(<expr>, …)**
  - **print(<expr>, …, end=<str>)**
- Are these allowed?
  - print("The answer is:", 7 * 3 * 2)
  - print(3, 5, 7, end="")
- Semantics?

A “slot” to be filled with any expression

Repeat indefinitely as many times as desired, zero or more

Q5

ROSE-HULMAN INSTITUTE OF TECHNOLOGY

Variables

- Identifiers (i.e.names) that refer to values stored in memory
- Values can be of any type

```
width = 4
temperature = 98.6

dogName = "fido"
lost = [4, 8, 15, 16, 23, 42]

triangleArea = width * height / 2
xyPoint = (r * cos(theta), r * sin(theta))
```

Variables and Assignment

- Assignment gives a variable a value

  x = 6 * 7

  – Python evaluates right-hand side (**42**)

  – Then variable on left “gets” the value

- “Gets” not “Equals”

  – **x = 3.9 * x * (1 – x)**

How to Think About Variables

- Variables as sticky notes
- Example on board…

```
x = 10

x = x + 1
```

Three Kinds of Assignment

- Simple
- Compound
- Multiple (or simultaneous)

Simple Assignment

- **<variable> = <expr>**

- Note:
  - **input(<string>)** is an expression
  - input statements are a kind of assignment statement

Q6,7

Compound Assignment

- **<var> <op=> <expr>** means
  <var> = <var> <op> <expr>
  - where **<op=>** is **+=**, **-=**, ***=**, **/=**, **//=**, or **%=**
- Example:
  - **total += 5** is the same as
    total = total + 5

Simultaneous Assignment

- **<var>, <var>, … = <expr>, <expr>, …**
- Example:
  - **sum, diff = x + y, x - y**

Assignment Assignment

- See **assignmentsAndLoops.py** module
- Do the TODOs inside the **assignmentStatements()** function

Summary: Assignment Statements

- Simple assignments: **<variable> = <expr>**
- Compound assignments
  - **<var> <op=> <expr>** means
    <var> = <var> <op> <expr>
    where **<op=>** is **+=**, **-=**, ***=**, **/=**, **//=**, or **%=**
- Simultaneous (multiple) assignments
  - **<var>, <var>, … = <expr>, <expr>, …**

Sequence

- A list of things
- For example:
  - **[2, 3, 5, 7]**
  - **[“My”, “dog”, “has”, “fleas”]**
- Every **for** loop uses a list

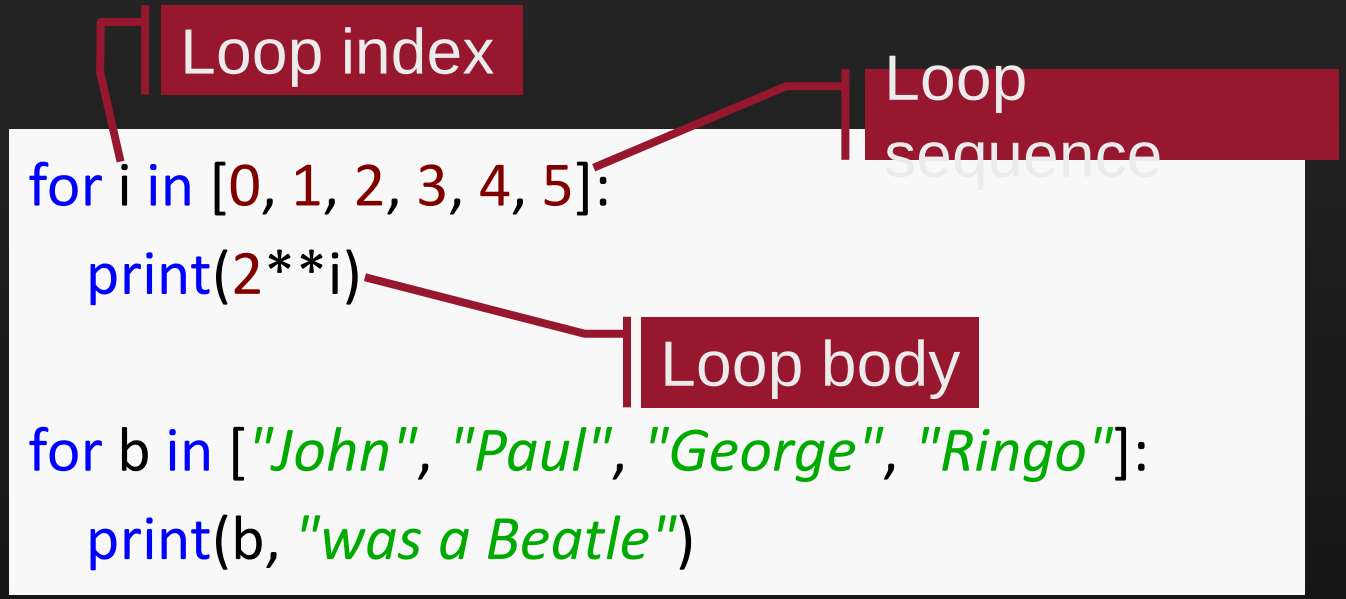
Definite Loops

- *Loop*: a control structure for executing a portion of a program multiple times
- *Definite loop*: Python knows beforehand how many times to repeat the body of the loop
- Syntax:

  for <var> in <sequence> :
  <body>

- Semantics: Executes **<body>** once for every element of **<sequence>**, with **<var>** set to that element.

Some Definite Loops



The range Function

- Creates a list that is an *arithmetic sequence*
- General formats for range function:
  - **range(<expr>)**
  - **range(<expr>, <expr>)**
  - **range(<expr>, <expr>, <expr>)**

- Consider:

```
list(range(8))
list(range(1, 7))
list(range(3, 18, 2))
list(range(4,10,-1))
list(range(17, -5, -3))
```

Q9

Work Time

HW2 due in 24 hours, HW3 due at start of next class

Q10-11

ROSE-HULMAN INSTITUTE OF TECHNOLOGY