Objectives for class 3

- --- Chapter 2 ---

2.9 To use augmented assignment operators to simplify coding (§2.11)

2.10 To perform numeric type conversion and rounding with the round function (§2.12)

2.11 To describe the software development process and apply it to develop the loan payment program (§2.14)

- --- Chapter 3---

3.1 To write Boolean expressions using relational operators (§3.2).

3.2 To generate random numbers using the random.randint(a, b), random.randrange(a, b), or random.random() functions (§3.3).

3.3 To program with Boolean expressions (AdditionQuiz) (§3.3).

3.4 To implement selection control using one-way if statements (§3.4).

3.5 To implement selection control using two-way if-else statements (§3.5).

How to Use Augmented Assignment Operators?

- **Combine** assignment and arithmetic operators.
  - **+=** *addition assignment operator*.
- Performed last after all the other operators on RHS are evaluated.

```
>>> j=1
>>> i=1
>>> i+=j+j*5
>>> i
7
```

count = count + 1 **Same as** count += 1

Operator	Name	Example	Equivalent
`+=`	Addition assignment	`i += 8`	`i = i + 8`
`-=`	Subtraction assignment	`i -= 8`	`i = i - 8`
`*=`	Multiplication assignment	`i *= 8`	`i = i * 8`
`/=`	Float division assignment	`i /= 8`	`i = i / 8`
`//=`	Integer division assignment	`i //= 8`	`i = i // 8`
`%=`	Remainder assignment	`i %= 8`	`i = i % 8`
`**=`	Exponent assignment	`i **= 8`	`i = i ** 8`

Type Determines the Type of Operations

- Python knows what "**type**" everything is
  - String,integer,float,list,tuple,dictionary
- **type** determines the operations
  - You cannot "add 1" to a string
- **type()** tells what type something is

```
>>> eee = 'hello ' + 'there'

>>> eee = eee + 1
Traceback (most recent call
last):  File "<stdin>", line
1, in <module>TypeError:
Can't convert 'int' object
to str implicitly

>>> type(eee)
<class'str'>
>>> type('hello')
<class'str'>
>>> type(1)
<class'int'>
>>> type(1.0)
< class'float'> >
```

Convert int to float or float to int

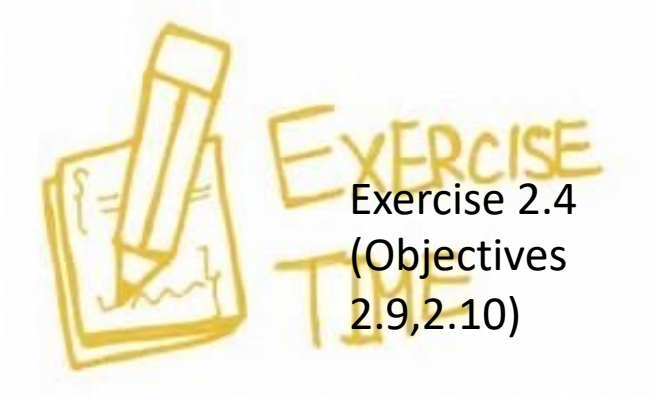
- The integer is **implicitly** converted to a float
- int() converts other types into integer
- float() convers other types into float

```
>>> print(99.0 + 100)
199.0
>>> print(float(99) + 100)
199.0
>>> i = 42
>>> type(i)
<class'int’>

>>> f = float(i)
>>> print(f)
42.0
>>> type(f)
<class'float'>
```

Convert a string into int or float

- Get an error if the string does not contain a numerical value



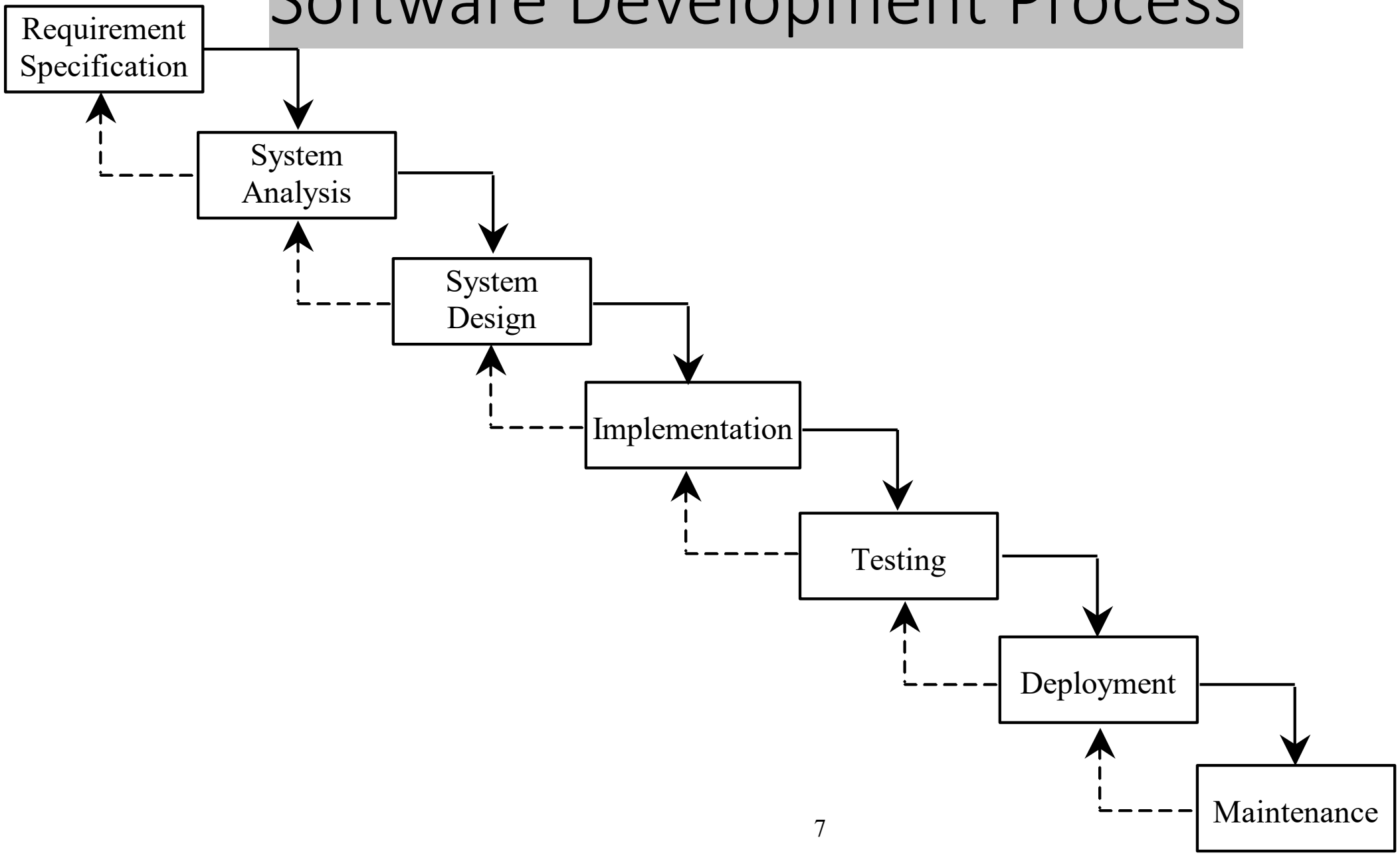
Exercise 2.4 (Objectives 2.9,2.10)

```
>>> sval = '123'
>>> type(sval)
<class 'str'>
>>> print(sval + 1)
Traceback (most recent call last):  File
"<stdin>", line 1, in <module>
TypeError: Can't convert 'int' object to str
implicitly

>>> ival = int(sval)
>>> type(ival)
<class 'int'>
>>> print(ival + 1)
124

>>> nsv = 'hello bob'
>>> niv = int(nsv)
Traceback (most recent call last):  File
"<stdin>", line 1, in <module>
ValueError: invalid literal for int() with
base 10: 'x'
```

Software Development Process



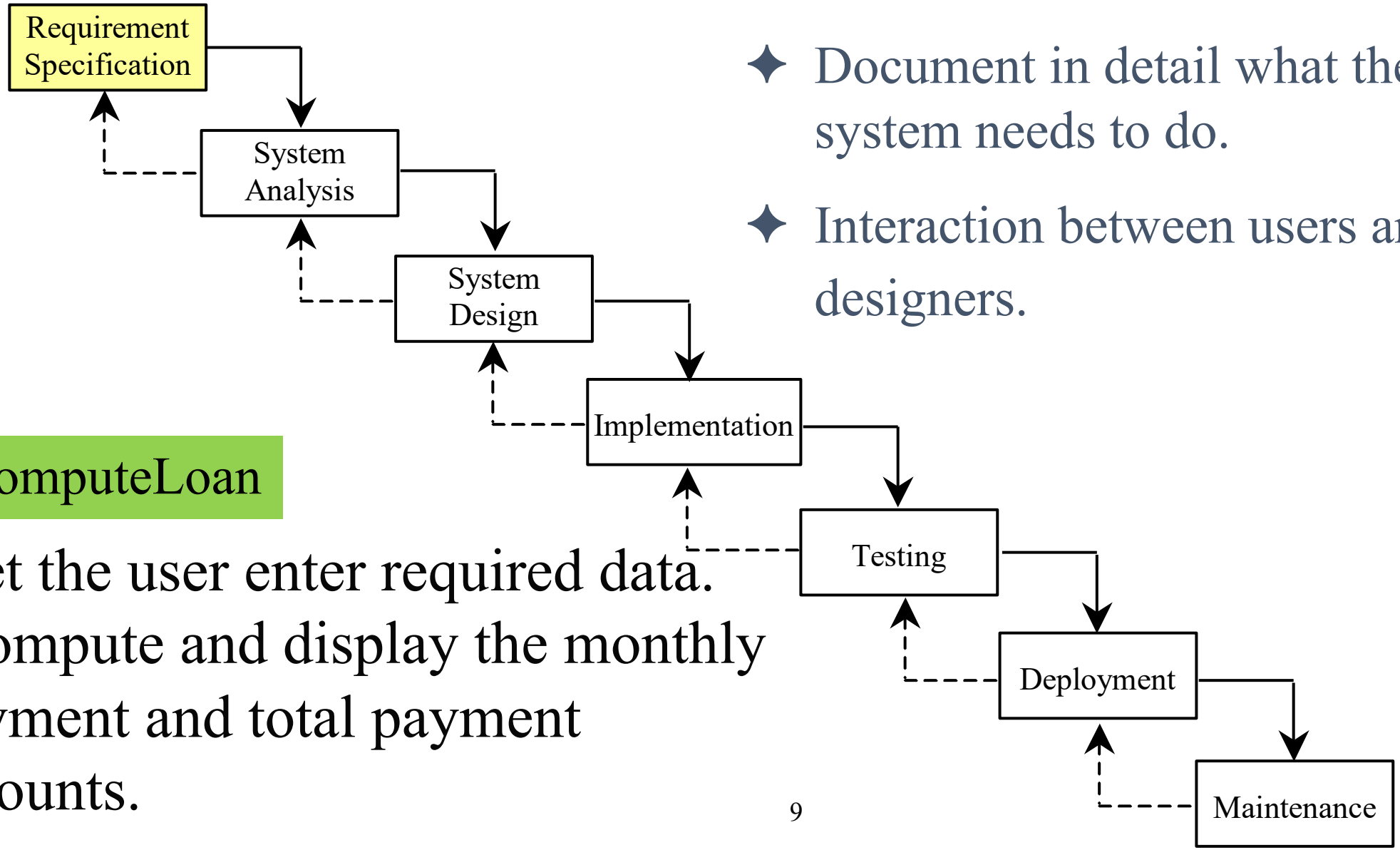
Problem: Computing Loan Payments

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

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

ComputeLoan

Requirement Specification : Document what the software needs to do

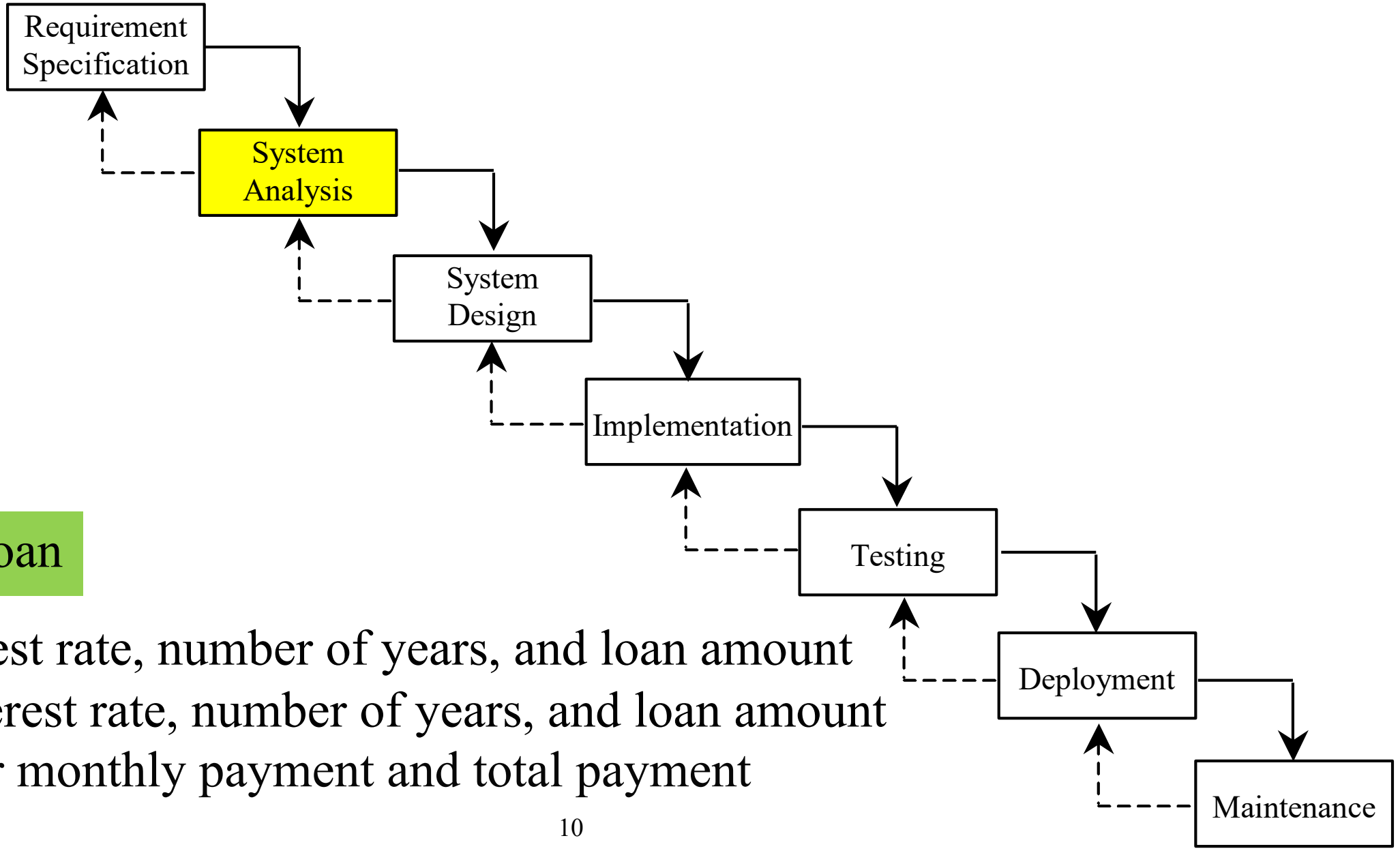


- Document in detail what the software system needs to do.
- Interaction between users and designers.

ComputeLoan

- Let the user enter required data.
- Compute and display the monthly payment and total payment amounts.

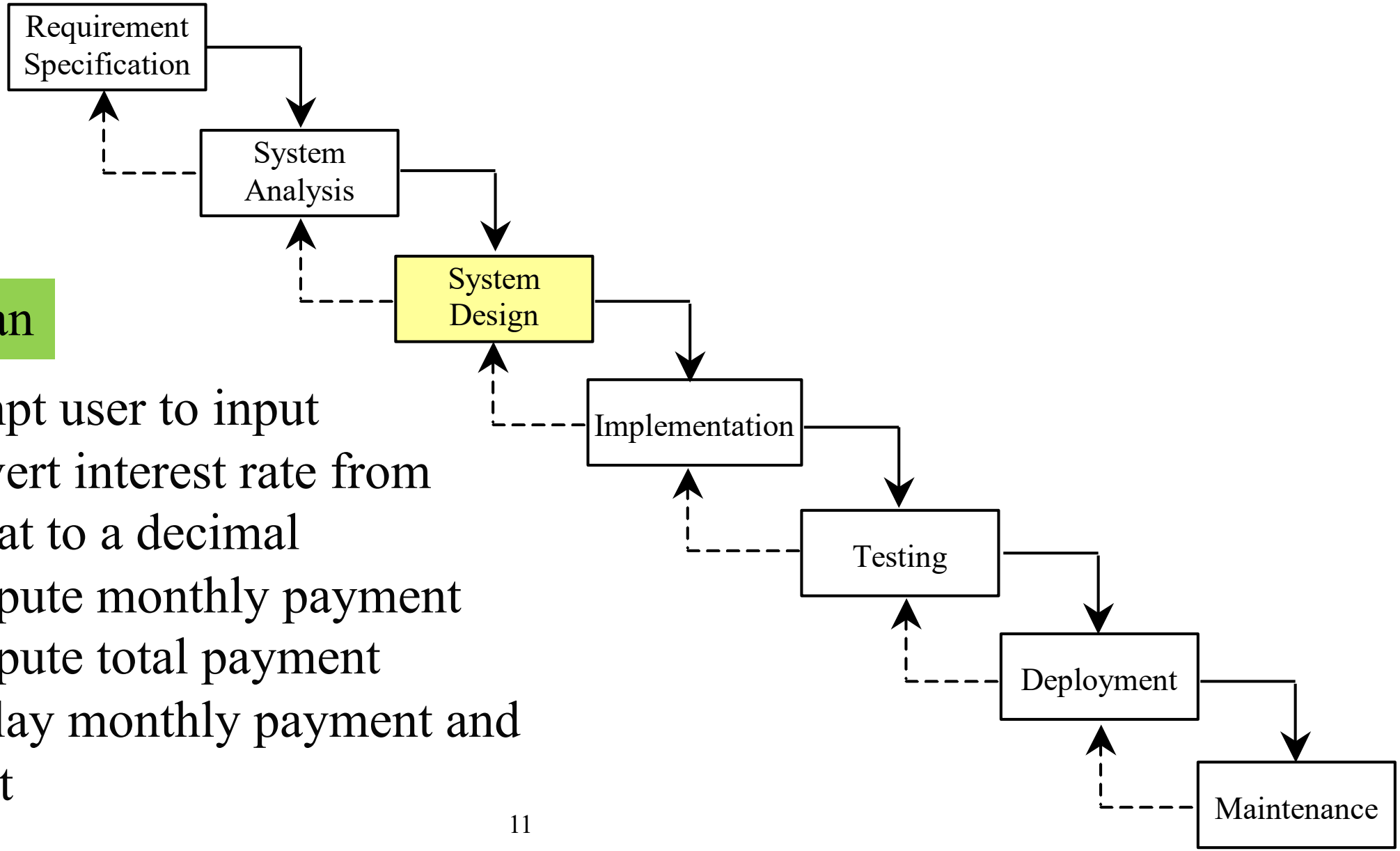
System Analysis : Define input and output



ComputeLoan

- Input: interest rate, number of years, and loan amount
- Output: interest rate, number of years, and loan amount
- Formula for monthly payment and total payment

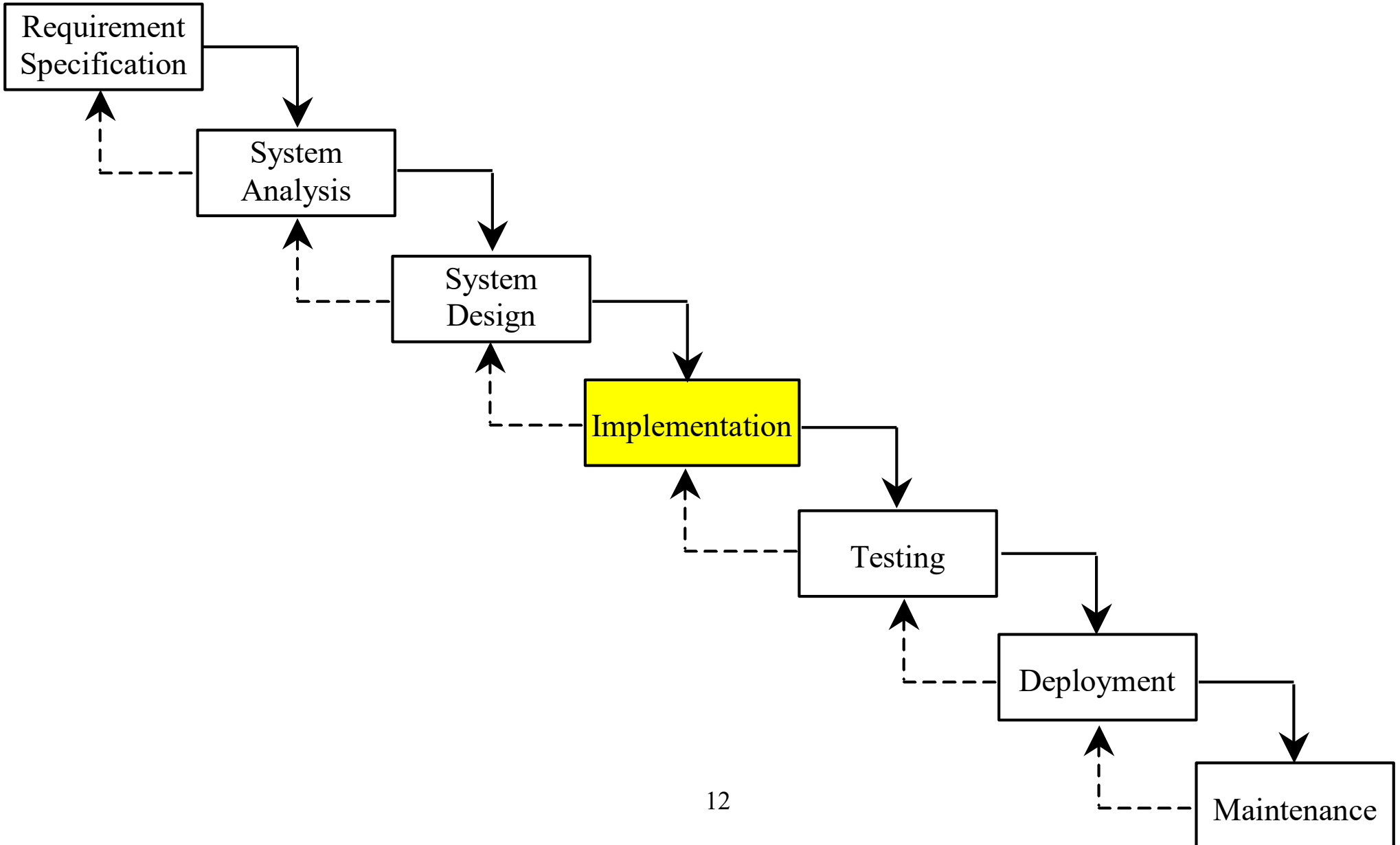
System Design : Steps from Input to Output



ComputeLoan

- Step1: Prompt user to input
- Step2: Convert interest rate from percent format to a decimal
- Step3: Compute monthly payment
- Step4: Compute total payment
- Step5: Display monthly payment and total payment

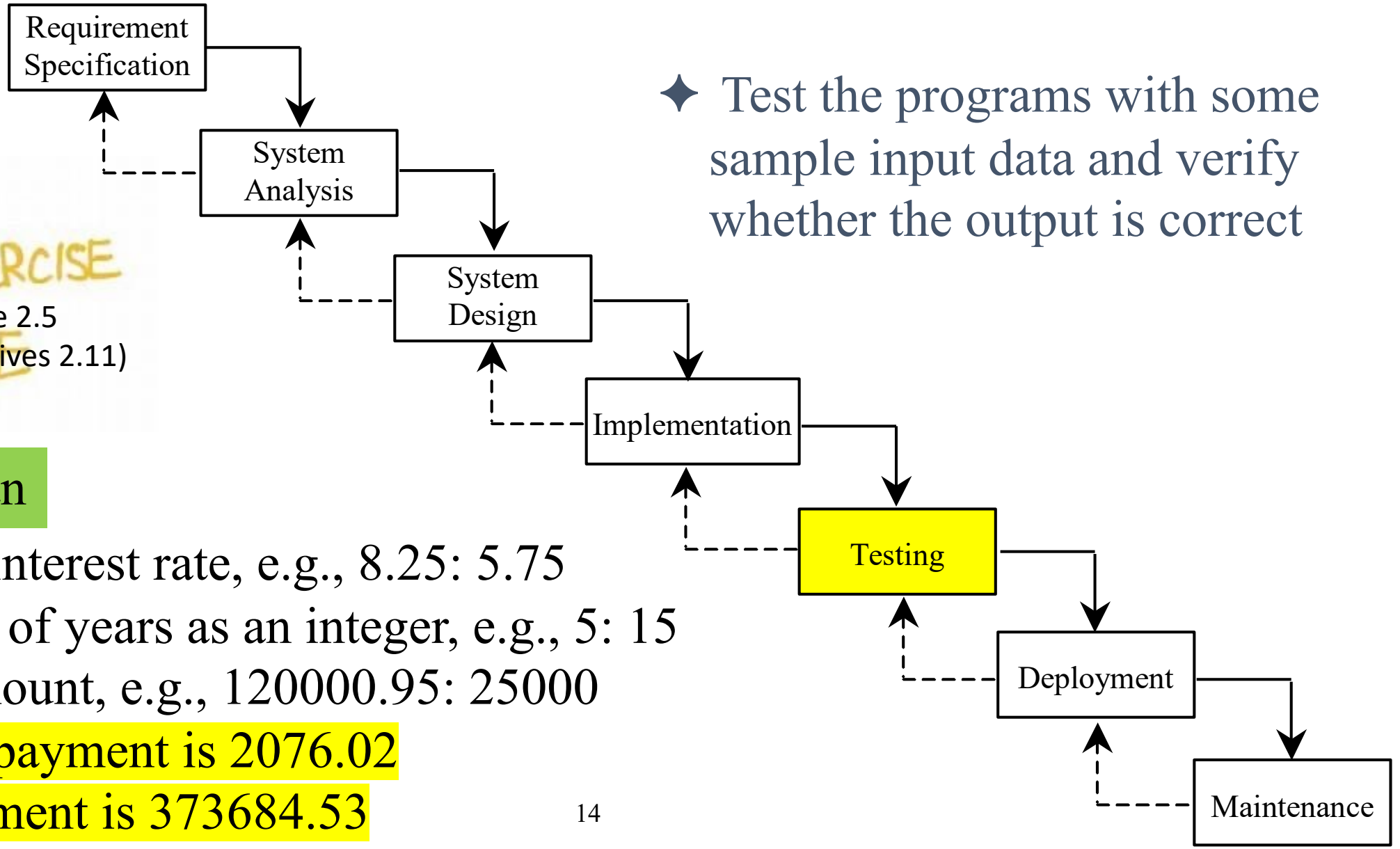
Implementation : Write Programs



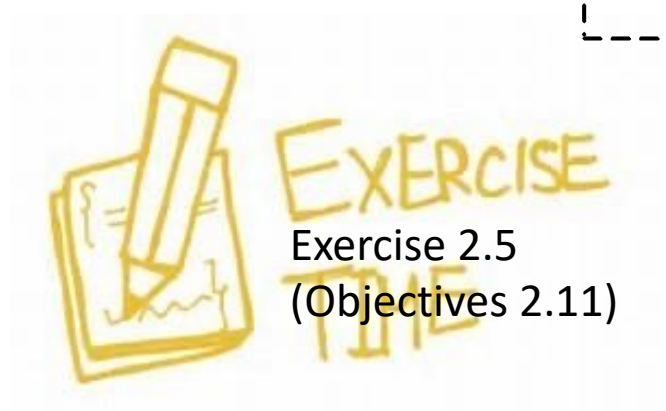
ComputeLoan

```
1  # Enter annual interest rate
2  annualInterestRate = float(input(
3    "Enter annual interest rate, e.g., 8.25: "))
4  monthlyInterestRate = annualInterestRate / 1200
5
6  # Enter number of years
7  numberOfYears = int(input(
8    "Enter number of years as an integer, e.g., 5: "))
9
10 # Enter loan amount
11 loanAmount = float(input("Enter loan amount, e.g., 120000.95: "))
12
13 # Calculate payment
14 monthlyPayment = loanAmount * monthlyInterestRate / (1
15   - 1 / (1 + monthlyInterestRate) ** (numberOfYears * 12))
16 totalPayment = monthlyPayment * numberOfYears * 12
17
18 # Display results
19 print("The monthly payment is", int(monthlyPayment * 100) / 100)
20 print("The total payment is", int(totalPayment * 100) /100)
```

Testing : Verify if Output Correct



- Test the programs with some sample input data and verify whether the output is correct



Exercise 2.5
(Objectives 2.11)

ComputeLoan

Enter annual interest rate, e.g., 8.25: 5.75
Enter number of years as an integer, e.g., 5: 15
Enter loan amount, e.g., 120000.95: 25000
The monthly payment is 2076.02
The total payment is 373684.53