Masters in QA Automation

Day #2

Introduction To Programming

To have

Notebook
Pen/Pencil
Water Bottle
Kerchief/Tissues

i ♥ programming

Are you Ready?

wish

Recap

6 Step Strategy – Programming Problems

1. Understand the problem
2. Design test data / test cases (input and expected output)

3. Derive the solution - solve the problem (writing pseudo code)
4. Test the solution (against the test data/case)

5. Write the program/code (using Java)
6. Test the code (syntax errors, run time errors, logical errors)

6 Step Strategy – Programming Problems

i♥programming

1. **Understand the problem**
2. **Design test data / test cases (input and expected output)**

3. Derive the solution - solve the problem (writing pseudo code)
4. Test the solution (against the test data/case)

5. Write the program/code (using Java)
6. Test the code (syntax errors, run time errors, logical errors)

Tips to understand the Problem

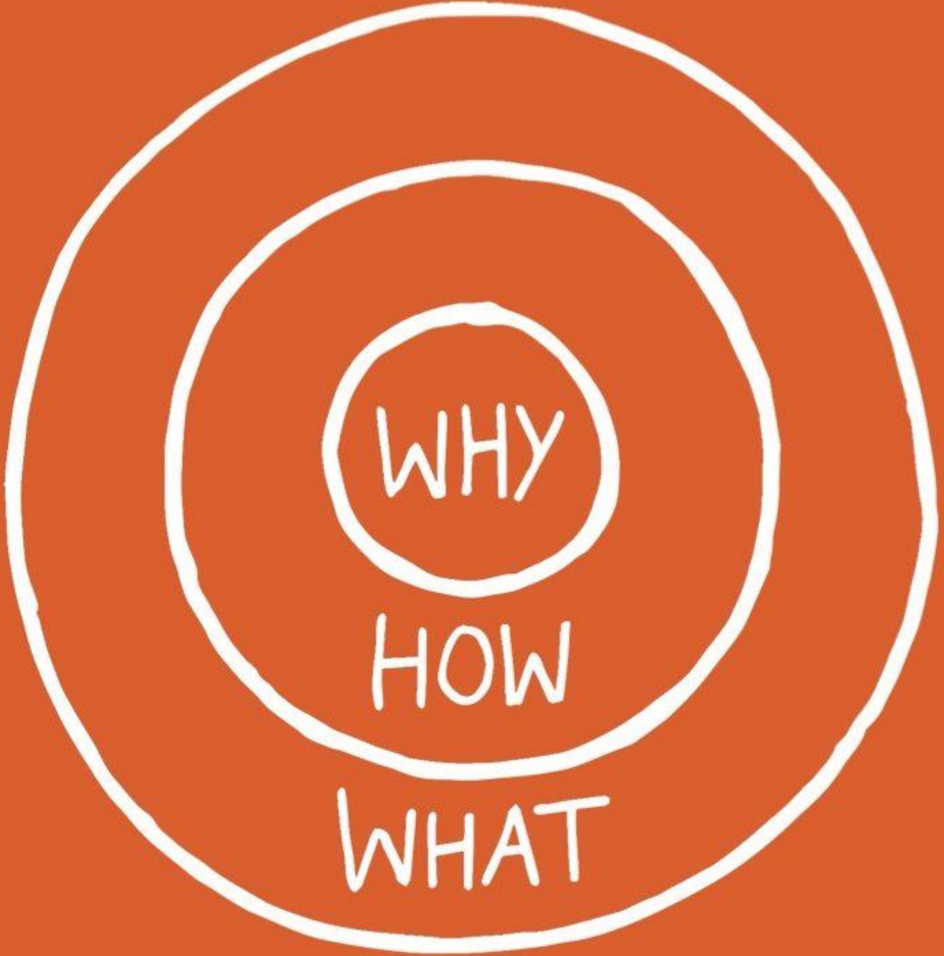
More than one
- Input
- Constraint

- Input (sample)
  - Valid
    - Simple
    - Harder/Difficult
  - Invalid

- Constraints (sample)
  - Time taken to execute
  - Less usage of space/memory
  - No inbuilt functions
  - Using or not using recursion

Bonus

The Golden Circle

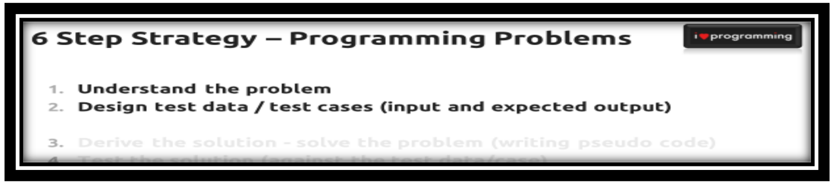


Take Home Assignments

Take Home Assignment

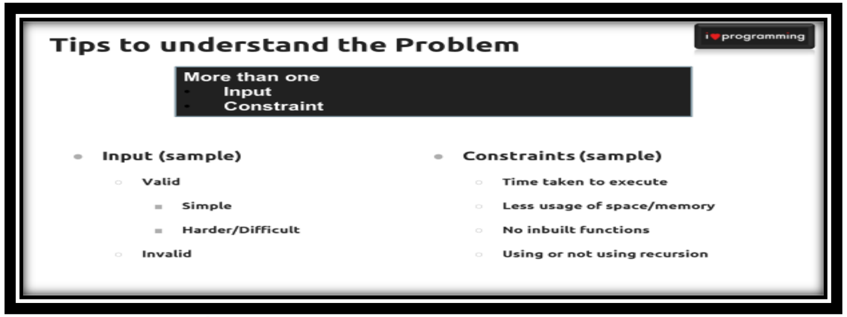
Apply Step 1 and 2 for the below problems

1. Find the sum of 2 numbers
2. Find the smallest of 3 numbers (Conditional and operators)
3. Classify input as odd or even (Operators)
4. Check if a given year is a leap year (EITHER The year is multiple of 400 OR the year is multiple of 4 and not multiple of 100.) (Conditionals)
5. Given a student's mark, decide their grade ([0, 25] - F, [25, 45] - E, [45, 50] - D, [50, 60] - C, [60, 80] - B, [80, 100] - A) (Conditionals)
6. Given a number n. Add all the numbers from 1 to n. (loop)
7. Check if the given number is a prime number (loop)
8. Find the nth fibonacci number (loop)
9. Find the sum of all elements in a Matrix (2D array concept)
10. Find if a given string is a palindrome (String operation)
11. Find if a string is an isogram (An isogram is a string that has no repeating character. Set concept)
12. Find the number of times each character is repeated in a string (Map concept)



6 Step Strategy – Programming Problems

1. Understand the problem
2. Design test data / test cases (input and expected output)
3. Derive the solution - solve the problem (writing pseudo code)
4. Test the solution (against the test data/case)



Tips to understand the Problem

More than one
- Input
- Constraint

- Input (sample)
  - Valid
    - Simple
    - Harder/Difficult
  - Invalid
- Constraints (sample)
  - Time taken to execute
  - Less usage of space/memory
  - No inbuilt functions
  - Using or not using recursion

Sample Template

i ♥ programming

Return sum of 2 numbers

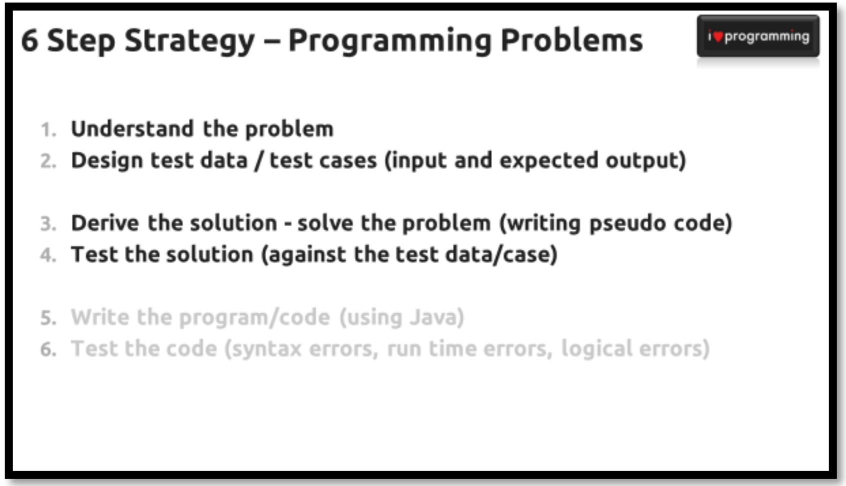
	Input	Output
Valid	2, 5	7
InValid	12, *	Invalid Input

List your Questions and Assumptions!

Solution for Take Home Assignments

Agenda

- Step 3 and 4 of **“6 Step Strategy – To Solve Programming Problems”**
- Bonus
  - Sure-shot tip for enhance your learning here!



6 Step Strategy – Programming Problems

i ❤ programming

1. Understand the problem
2. Design test data / test cases (input and expected output)
3. Derive the solution - solve the problem (writing pseudo code)
4. Test the solution (against the test data/case)
5. Write the program/code (using Java)
6. Test the code (syntax errors, run time errors, logical errors)

Let's Start Coding!

Problem #3:

Write A Program

Return sum of 2 numbers

11, 3 = 11+3 = 14

Let's Start Understand!

Problem #3:

Write A Program

Return sum of 2 numbers

11, 3 = 11+3 = 14

Let's Start Understand!

i ♥ programming

Problem #3:

Write A Program

Return sum of 2 numbers

11, 3 = 11+3 = 14

CAN YOU PUT IT IN THE CHAT?

6 Steps Strategy – Programming Problems

1. Understand the problem
2. Design test data / test cases (input and expected output)

3. Derive the solution - solve the problem (writing pseudo code)
4. Test the solution (against the test data/case)

5. Write the program/code (using Java)
6. Test the code (syntax errors, run time errors, logical errors)

Lets Work!

Write A Program

To Find Sum of 2 Numbers!

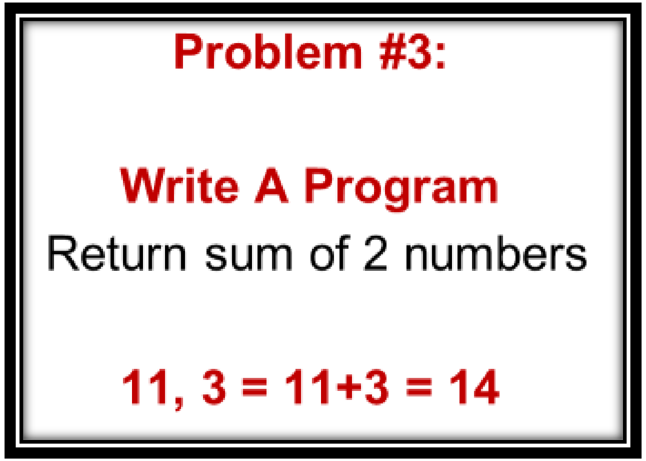
Input	2 Numbers. 4 digit number, -ve to +ve, Whole and Decimal number with one decimal point
Output	Number, which represents the sum of given 2 numbers

Category	Input	Output
Valid	123, 121	244
	-12, 39.5	27.5
	-12, -12	-24
	0, 0	0
	0, -121	-121
	2222, 9989	12211
	123.4, 56.3	179.7
	0, 9998.5	9998.5
	-9999, -9989	-19988
Invalid	99999, -999	"Invalid input, Please try with valid input"
	-993499, 12	"Invalid input, Please try with valid input"
	123.123, -1	"Invalid input, Please try with valid input"
	blank, 123	"Invalid input, Please try with valid input"
	alphanumeric, 292	"Invalid input, Please try with valid input"
	SpecialChar, 234	"Invalid input, Please try with valid input"

Derive the solution - solve the problem (writing pseudo code)

Tip: Step by Step Instructions to a dumb servant

1. Get 1st Number
2. Get 2nd Number
3. Add 1st and 2nd Number
4. Save the result
5. Print the result



Problem #3:

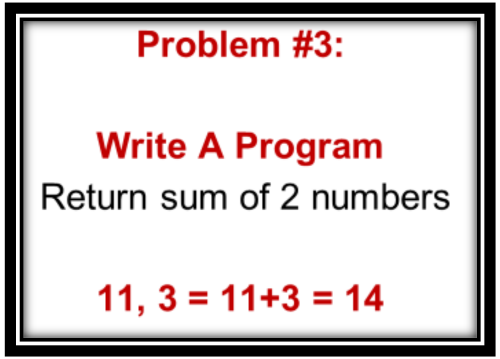
Write A Program

Return sum of 2 numbers

11, 3 = 11+3 = 14

Derive the solution - solve the problem (writing pseudo code)

Tip: Step by Step Instructions to a dumb servant


Problem #3:

Write A Program

Return sum of 2 numbers

11, 3 = 11+3 = 14

1. Get 1st Number (using a method/function in Java)
2. Get 2nd Number (using a method/function in Java)
3. Add 1st and 2nd Number (using addition + operator)
4. Save the result (using assignment = operator)
5. Print the result (using a method/function in Java)

Derive the solution - solve the problem (writing pseudo code)

Tip: Step by Step Instructions to a dumb servant

1. Get 1st Number (using a **method/function** in Java) and save it as **numberOne**
2. Get 2nd Number (using a **method/function** in Java) and save it as **numberTwo**
3. Add 1st and 2nd Number (using addition + operator)
4. Save the result (using a **method/function** in Java) as resultSum
5. Print the result (using a **method/function** in Java)

**during the next few weeks, you will get introduced to different methods/function using which you can achieve/perform specific tasks like printing, getting input, etc…*

Let's Start Solving!

Problem #3:

Write A Program

- To find if the given number is prime or not

4 digit

Only 4 Digit Number

To find if the given number is prime or not

- Get the input number
- **If the input is not a number, then**
  - Print “Invalid Input. Retry with valid input”
  - Exit
- If the input is not in the range of -9999 to 9999, **then**
  - Print “Invalid Input. Retry with valid input”
  - Exit
- If the given input is not whole number,
  - then Round off
  - proceed
- If the given input is valid negative number -1 to -9999, then
  - multiply with -1 (to convert the negative number to positive number)
  - Proceed

To find if the given number is prime or not

- Divide the number by each of the number in this list (2, 3, 4, till number-1)
    - If the reminder is zero, then print “Not A Prime” and Exit
- Print “Prime”
- Exit

Let's Start Solving!

Problem #2:

Write A Program
To print the sum of digits of a given input

123 = 1+2+3 = 6

To print the sum of digits of a given input

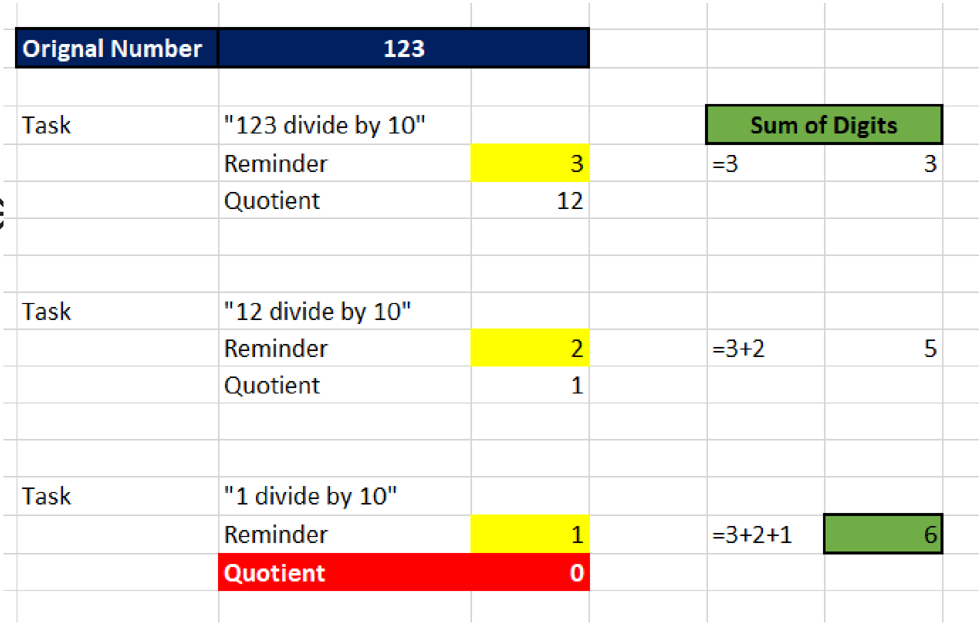
1. Get the input number
2. Split each digit from the number
    1. Use mathematical operations like division, addition and assignment
3. Add each digit and save the result
4. Print the sum

To print the sum of digits of a given input

1. Get the input number
2. **If the input is not a number, then**
    1. print “Invalid Input. Retry with valid input”
    2. exit
3. If the input is not in the range of -999999 to 999999, **then**
    1. print “Invalid Input. Retry with valid input”
    2. exit
4. If the input is not a whole number, **then**
    1. print “Invalid Input. Retry with valid input”
    2. exit

To print the sum of digits of a given input

5. If the given input is valid negative number -1 to -999999, then
   - multiply with -1 (to convert the negative number to positive number)
6. If the given input is single digit (between 0 to 9) then
   - Print the input number as is (3)
   - exit
7. Split each digit from the number as shown →
8. Add each digit
9. Print the sum



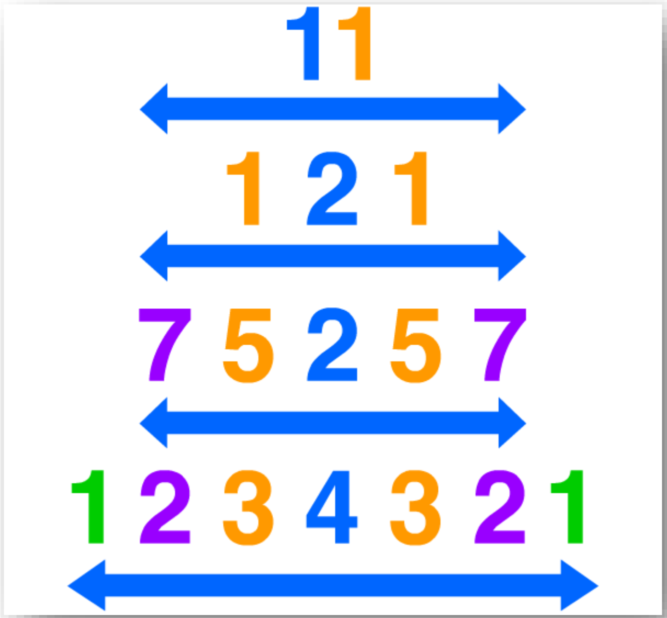
Orignal Number	**123**				
Task	"123 divide by 10"			**Sum of Digits**	
	Reminder	3		=3	3
	Quotient	12			
Task	"12 divide by 10"				
	Reminder	2		=3+2	5
	Quotient	1			
Task	"1 divide by 10"				
	Reminder	1		=3+2+1	6
	Quotient	**0**			

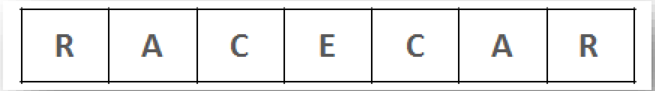
Let's Start Coding!

Problem #3:

Write A Program
To find if the given input is a palindrome or not?!

Palindrome or Not?





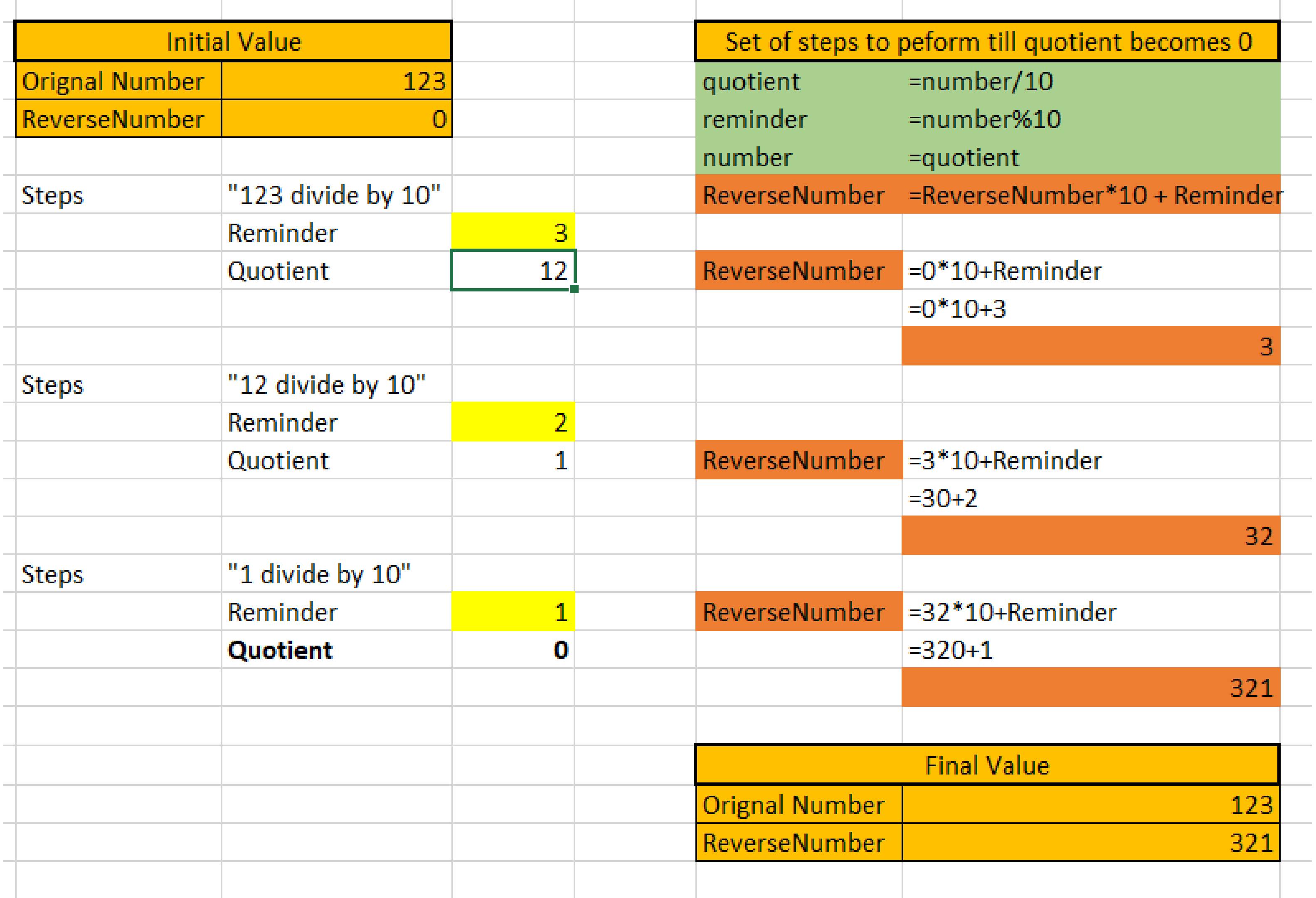
R	A	C	E	C	A	R

Given Input Number - Palindrome or not?

1. Get the input number
2. **If the input is not a number, then**
    1. print “Invalid Input. Retry with valid input”
    2. exit
3. If the input is not in the range of -999999 to 999999, **then**
    1. print “Invalid Input. Retry with valid input”
    2. exit
4. If the input is not a whole number, **then**
    1. print “Invalid Input. Retry with valid input”
    2. exit

Given Input Number - Palindrome or not?

5. If the given input is valid negative number -1 to -999999, then
   - multiply with -1 (to convert the negative number to positive number)
6. If the given input is single digit (between 0 to 9) then
   - Print the input is a palindrome
   - exit
7. Split each digit from the number and calculate the reverse of the input number *(refer next slide)*
8. Compare input number and reverse number
   - If equal, then "Input is Palindrome"
   - If not equal, then "Input is NOT a Palindrome"



Initial Value	
Orignal Number	123
ReverseNumber	0

Steps	"123 divide by 10"	
	Reminder	3
	Quotient	12
Steps	"12 divide by 10"	
	Reminder	2
	Quotient	1
Steps	"1 divide by 10"	
	Reminder	1
	Quotient	**0**

Set of steps to peform till quotient becomes 0	
quotient	=number/10
reminder	=number%10
number	=quotient
ReverseNumber	=ReverseNumber*10 + Reminder

ReverseNumber	=0*10+Reminder
	=0*10+3
	3
ReverseNumber	=3*10+Reminder
	=30+2
	32
ReverseNumber	=32*10+Reminder
	=320+1
	321

Final Value	
Orignal Number	123
ReverseNumber	321

6 Step Strategy – Programming Problems

1. **Understand the problem**
2. **Design test data / test cases (input and expected output)**

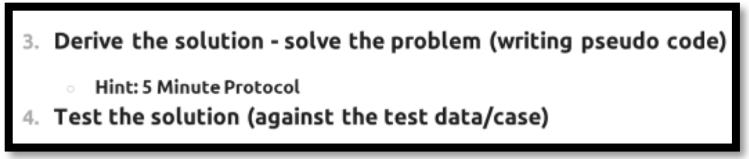
3. **Derive the solution - solve the problem (writing pseudo code)**
    - **Hint: 5 Minute Protocol**
4. **Test the solution (against the test data/case)**

5. **Write the program/code (using Java)**
6. **Test the code (syntax errors, run time errors, logical errors)**

i ❤ programming

Take Home Assignment

Apply Step 3 and 4 (of 6 step strategy) for the below problems

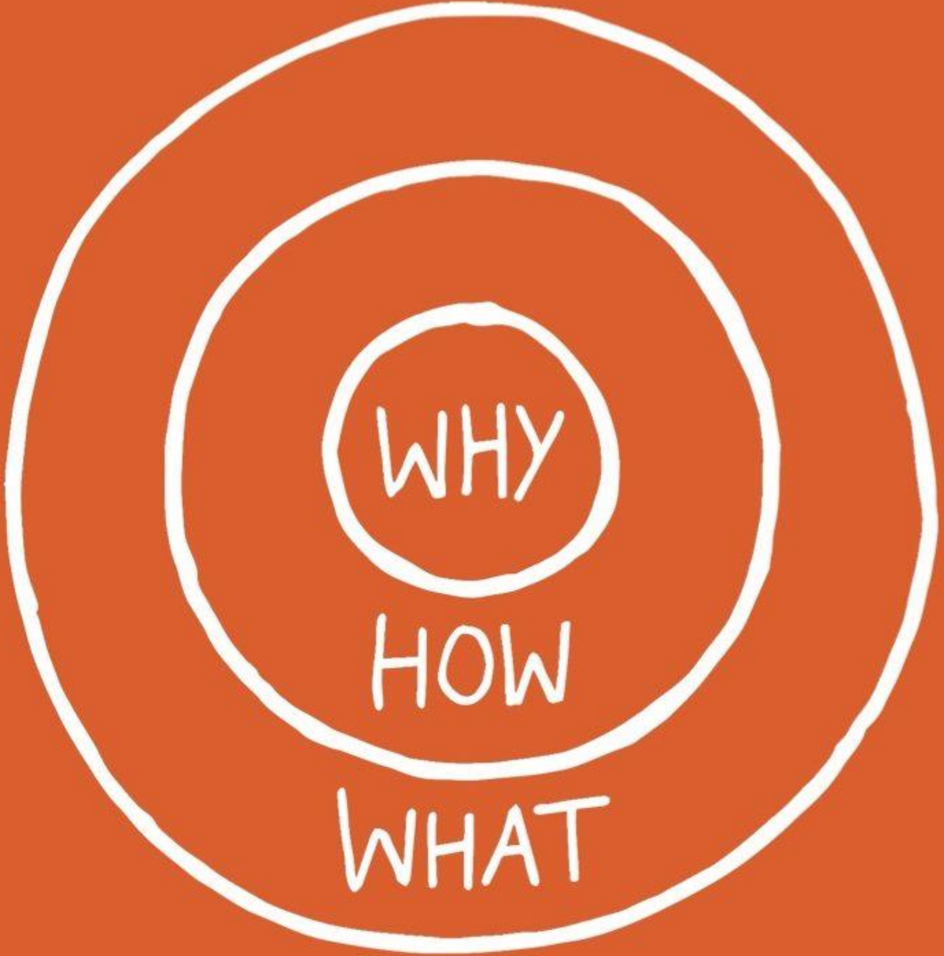


3. **Derive the solution - solve the problem (writing pseudo code)**
   - Hint: 5 Minute Protocol
4. **Test the solution (against the test data/case)**

1. Find the sum of 2 numbers
2. Find the smallest of 3 numbers (Conditional and operators)
3. Classify input as odd or even (Operators)
4. Check if a given year is a leap year (EITHER The year is multiple of 400 OR the year is multiple of 4 and not multiple of 100.) (Conditionals)
5. Given a student's mark, decide their grade ([0, 25] - F, [25, 45] - E, [45, 50] - D, [50, 60] - C, [60, 80] - B, [80, 100] - A) (Conditionals)
6. Given a number n. Add all the numbers from 1 to n. (loop)
7. Check if the given number is a prime number (loop)
8. Find the nth fibonacci number (loop)
9. Find the sum of all elements in a Matrix (2D array concept)
10. Find if a given string is a palindrome (String operation)
11. Find if a string is an isogram (An isogram is a string that has no repeating character. Set concept)
12. Find the number of times each character is repeated in a string (Map concept)

Bonus

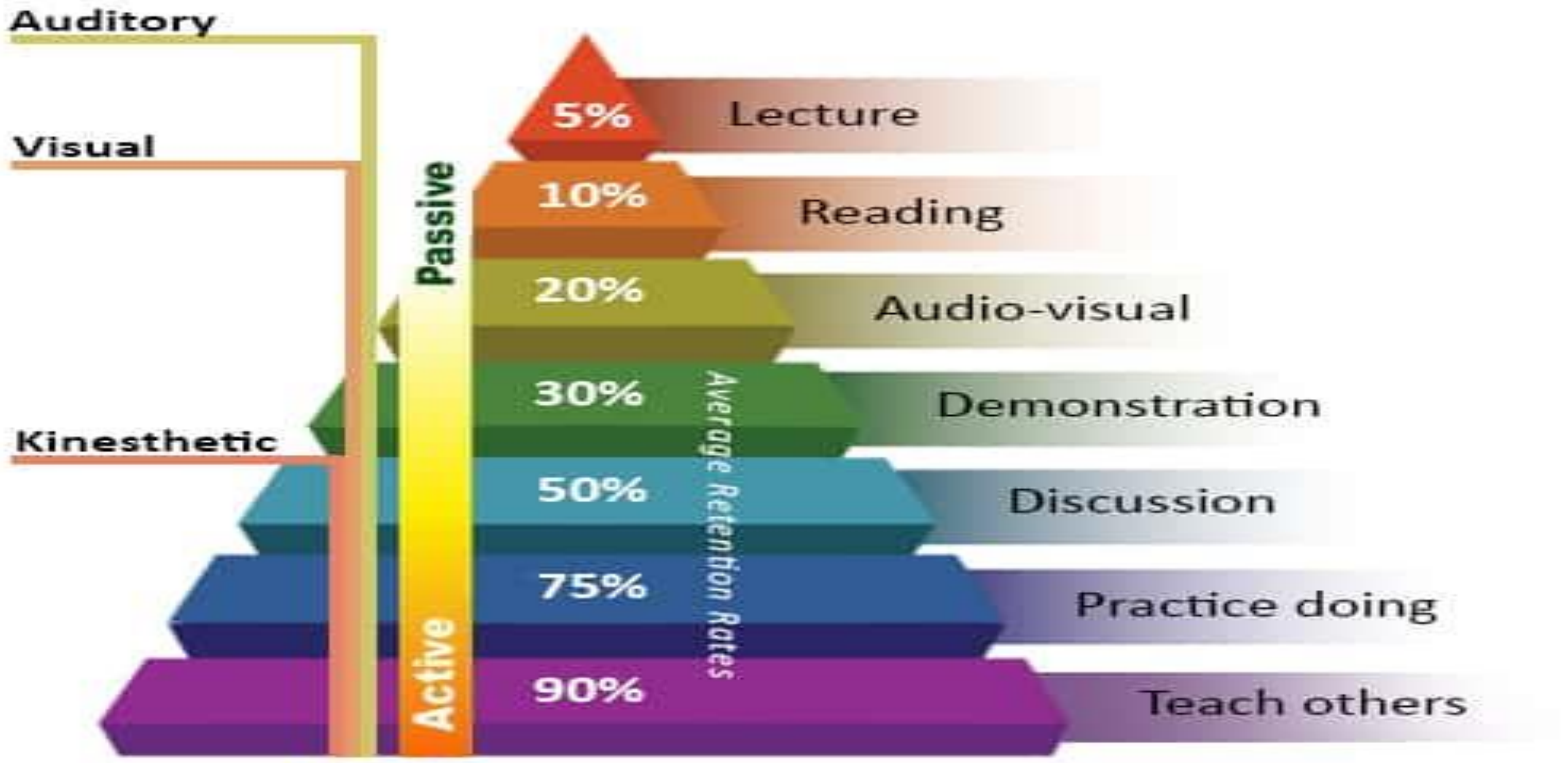
The Golden Circle



Bonus

Mode of acquiring knowledge

- Lecture Sessions
- Reading
- Listening to audio
- Watching Videos
- Seeing Demonstrations
- Discussions / Interviews
- Practical Experience (by doing)
- Teach Others!



Adapted from the NTL Institute of Applied Behavioral Science Learning Pyramid

See you by 7:30 PM Sunday