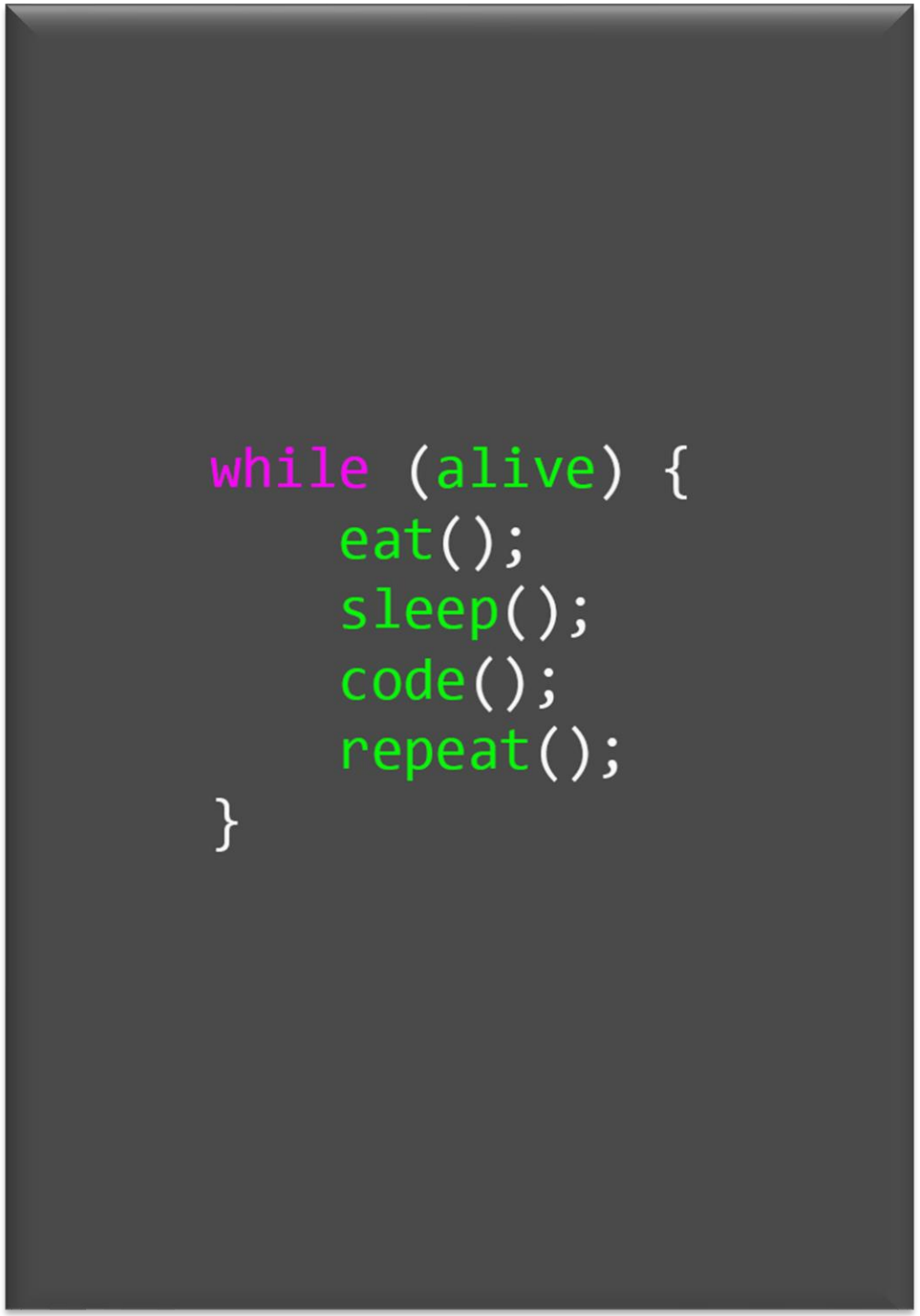


The Art of Programming

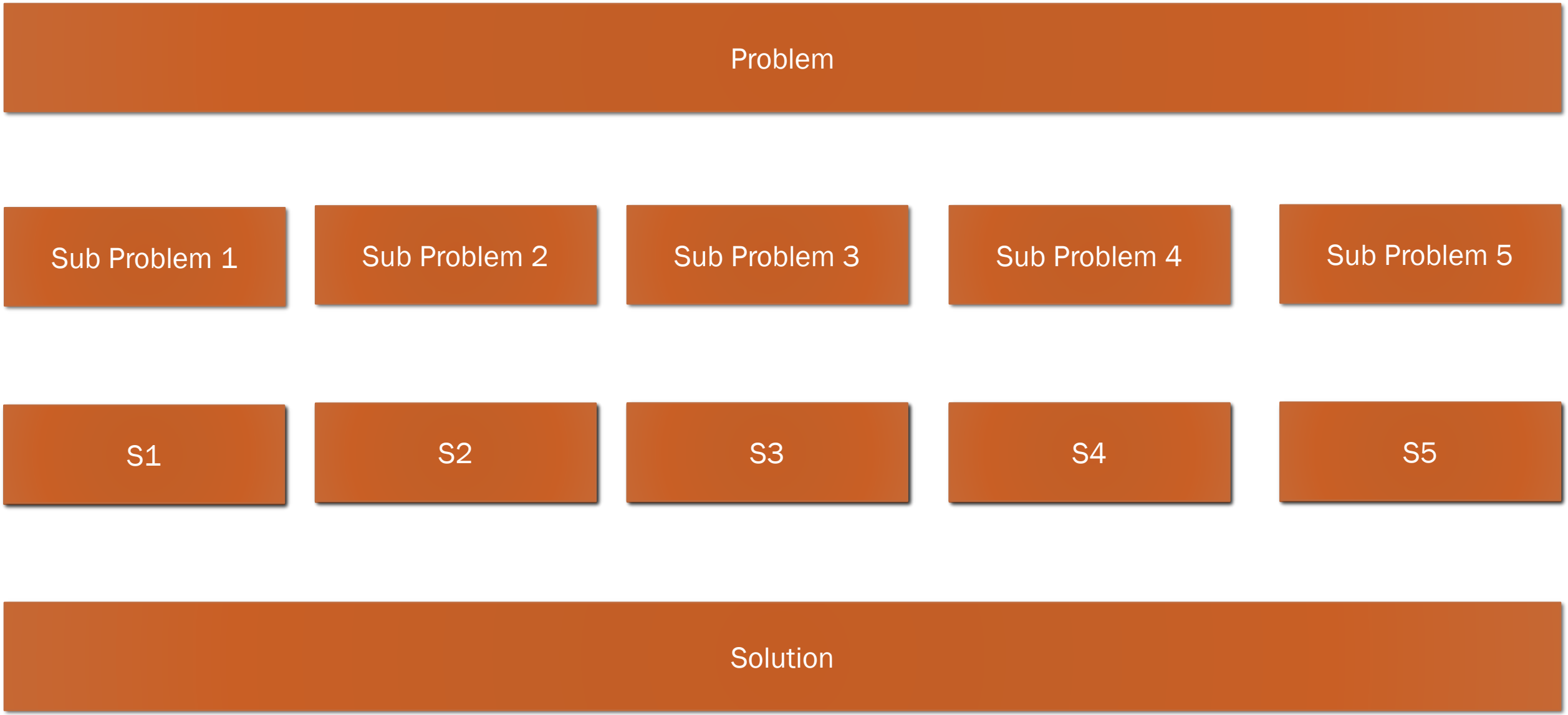
- KARUNA K B
- RAHUL JAIN

Intro

- Why Programming is an “***Art***” ?
- What it means to be a Programmer
- Programming – The Reality

Lesson 1: Problem Solving

- Definition.
- Importance.
- Why it is more of an **Art** than **Skill.**
- Role in programming
- Illustration


Problem
Sub Problem 1
Sub Problem 2
Sub Problem 3
Sub Problem 4
Sub Problem 5
S1
S2
S3
S4
S5
Solution

Print a Fibonacci triangle

Get the limit

Print Fibonacci series N times

Find Fibonacci upto a given number

Print each progressive series

Input n

start a loop from 1 to the limit given

print fibonnacci digits with a tab till limit terms

print

```
1
1 1
1 1 2
1 1 2 3
1 1 2 3 5
```

Lesson 2: Translation to Code

- Pseudocode
- Algorithm
- Tears (T_T)

Pseudocode (!Algorithm)

- Human understandable language
- Plain and simple solution to a problem

The Fibonacci Triangle

1. Read the limit from the user
2. Start a loop from 1 to the limit given
3. Print Fibonacci digits with a tab within loop till limit terms

Algorithm (a.k.a Magic)

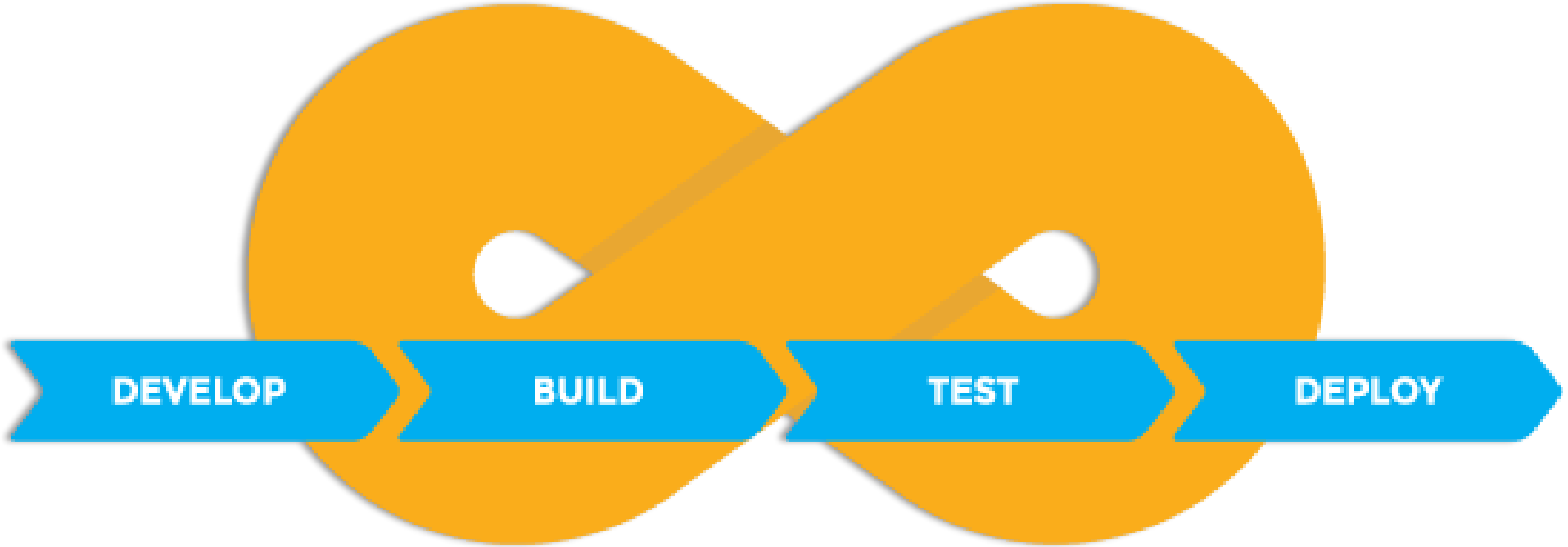
- Logical representation of a Pseudocode.
- Step-wise approach to solving a problem.
- DOES NOT REQUIRE KNOWLEDGE OF A PROGRAMMING LANGUAGE.

The Fibonacci Triangle

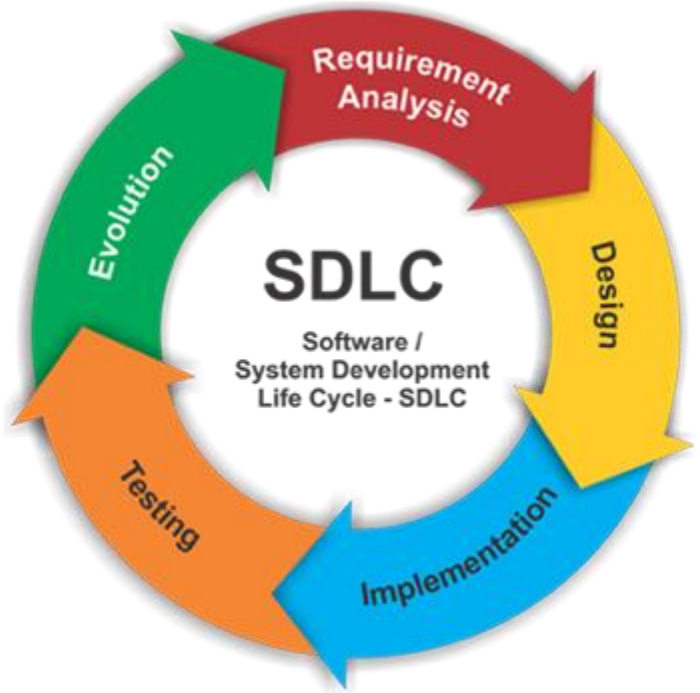
```
1. Start
2. Read n
3. for i = 1 to n then
4.       a = 1, b = 1
5.       for j = 0 to i then
6.           if j = 0 then
7.               print a
8.           else if j = 1 then
9.               print a, b
10.          else
11.              c = b + a
12.              print c
13               a = b
14.              b = c
15.          End Else-if
16.      End For
17. End For
18. Stop
```

```
1
1       1
1       1       2
1       1       2       3
1       1       2       3       5
1       1       2       3       5       8
1       1       2       3       5       8       13
1       1       2       3       5       8       13      21
1       1       2       3       5       8       13      21      34
```

Where does Code fit in?

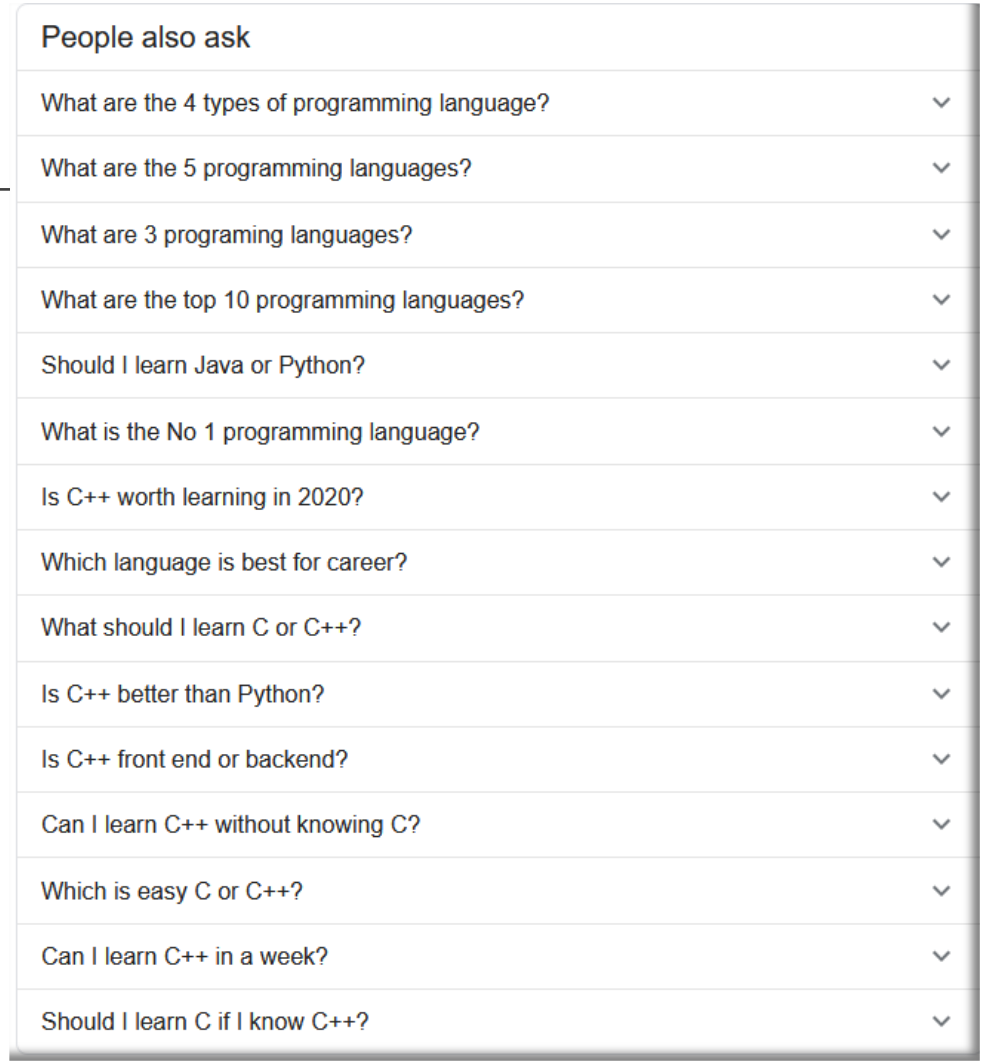


Corporate Coding Cycle - SDLC



The Great Dilemma!

- Which Programming Language to choose ?
- Which is the Best Programming Language ?



Lesson 3: Code Quality

- IDEs
- Techniques & Quality Analysis:
  - Reusability
  - Elegance
  - Modularity
  - Readability
  - Maintainability
  - Time – Complexity
  - SLOCs
  - The GOD Class

IDEs

- Language Deper
- Personal choice
- Corporation enfo

eclipse

IDEA

Beans.IDE

Tips for Naming

- Naming should express it's intent as clearly as possible. Change it when you find better.
- Try to create names distinctly expressing the context of the very intent
  - userID, getLargestString, findHighest, firstName, sampleKitValue
- User pronounceable and searchable names. Distinct names are easy to search in IDE
  - Avoid using generic names for variables, ex: acc1, acc2, x, y, i, j, etc
- Avoid mental mapping while naming.
  - Ex: acc for account, cus1 for firstCustomer
- Use noun for entity structure names, such as classes. Avoid using verbs here.
  - Customer, Vehicle, Sample, Company etc

Modularity & Reusability

```
sumOfSequares(a, b)
    squareOfA = getSquare(a)
    squareOfB = getSquare(b)
    sum(squareOfA, squareOfB)
```

```
getSquare(number)
    return (number * number)
```

```
sum(firstNumber, secondNumber)
    return firstNumber + secondNumber
```

Time Complexity

```
1. Start
2. Read n
3. for i = 1 to n then
4.       a = 1, b = 1
5.       for j = 0 to i then
6.           if j = 0 then
7.               print a
8.           else if j = 1 then
9.               print a, b
10.          else
11.              c = b + a
12.              print c
13               a = b
14.              b = c
15.          End Else-if
16.      End For
17. End For
18. Stop
```

vs

```
Start
input n
let fiboString = '';
let a=1;
let b=1;
for i from 0  to n then
    if i is 0 then
            fiboString += (a + '\t')
            print fiboString
        end-if
        else if i is 1 then
            fiboString += (a + '\t')
            print fiboString
        end else-if
        else
            let c= a+b;
            a=b;
            b=c;
            fiboString +=    (c + '\t')
            print fiboString
    end else
end for
end
```

Lesson 4: Optimizations

- Reinventing the wheel
- KISS – Keep It Simple Silly!
- Parallelization
- Scaling
- Space Complexity ?

Examples

```
Start
input n
let fiboString = '';
let a=1;
let b=1;
for i from 0  to n then
    if i is 0 then
            fiboString += (a + '\t')
            print fiboString
        end-if
        else if i is 1 then
            fiboString += (a + '\t')
            print fiboString
        end else-if
        else
            let c= a+b;
            a=b;
            b=c;
            fiboString +=    (c + '\t')
            print fiboString
    end else
end for
end
```

```
Start
Input n
arrayOfOnetoNineteen = ["Zero", "One", "Two" ..... "Nineteen"]
tens = { "", "", "twenty", ..... , "ninety" };
If number <20 then
        arrayOfOnetoNineteen[number]
Else
        ten = number/10;
        One = number % 10;
        If one is 0
                Print tens[ten]
        Else
                Print tens[ten] + arrayOfOnetoNineteen[one]
        End If-Else
End If-Else
End
```

Pit falls of Optimization

- Pre - Mature Optimization
- Optimization at the cost of readability
- Over - Optimization
- Optimization vs Reliability
- When to Optimize?

Conclusions - 1

- Nature
- Accessibility of Framework or Resources
- Will Coding become Obsolete ?

Conclusions - 2

- Stereotypes
  - Programmer
  - Programming
  - Technology
- How to master the “Art of Programming” ?

Questions?

Ask away because…

Thanks!

RAHUL JAIN | RJAIN.RAHUL5@GMAIL.COM | LINKEDIN

KARUNA K B | HEREISKKB@GMAIL.COM | LINKEDIN

Hakuna Matata!