SLIIT

Discover Your Future

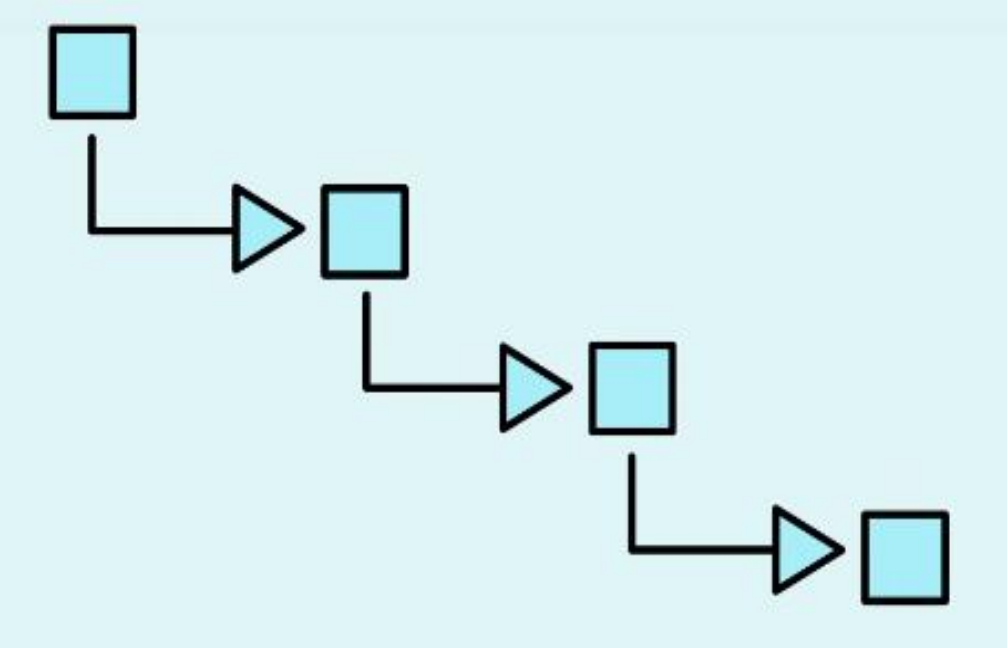
Paradigm Independent Metrics

Cognitive Functional Size (CFS) Measure

- Is a function of three fundamental factors:
  - Cognitive weights of Basic Control Structures (BCSs)
  - Number of inputs (Ni)
  - Number of outputs (No)

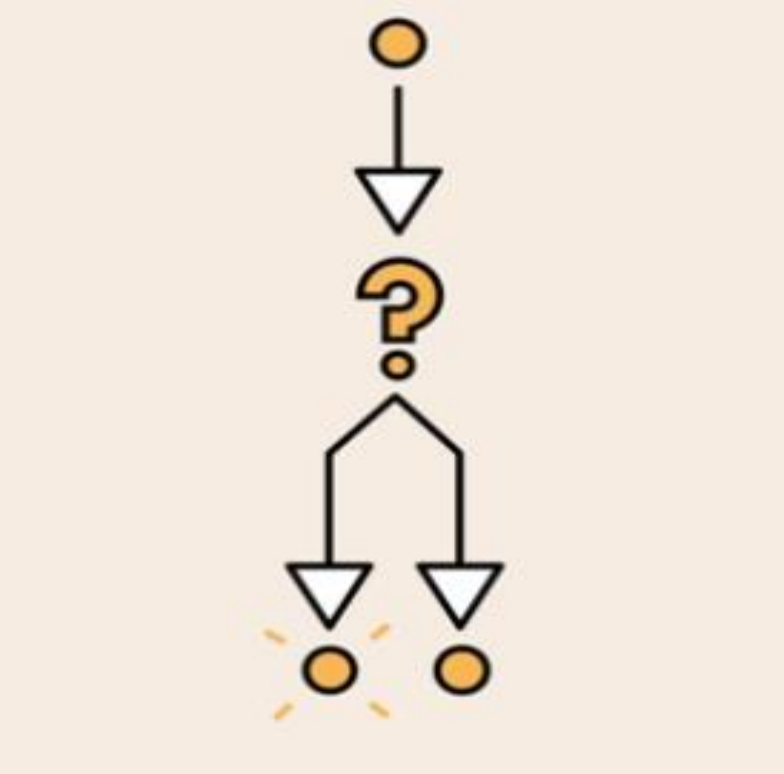
- The **cognitive weight** of software is the degree of difficulty or relative time and effort required for comprehending a given piece of software modelled by a number of BCSs.

Sequence Structures



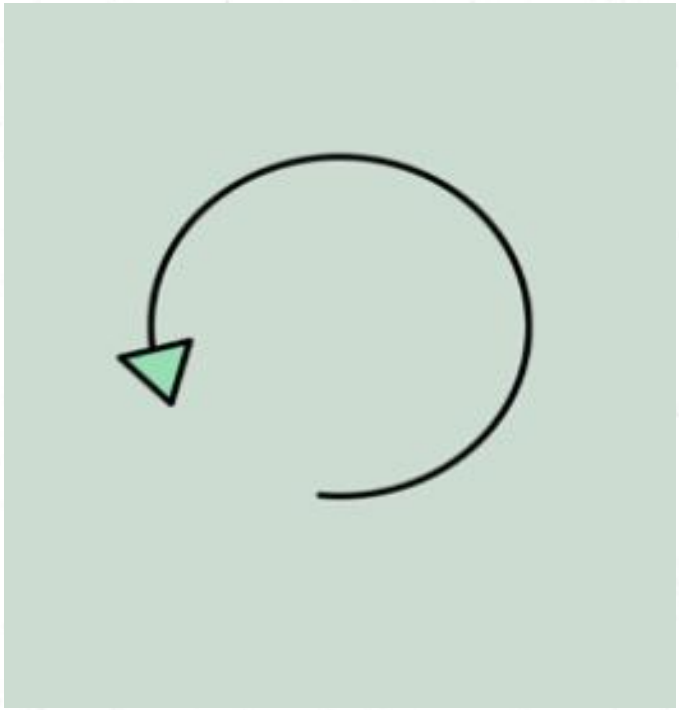
A sequence structure consists of a series of actions that is completed in a specific order. Action 1 is performed, then Action 2, then Action 3, etc., until all the actions in the sequence have been carried out.

Branch Structures



A branch or selection structure executes a certain piece of code only when a certain condition is met.

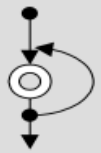
Iterative Structures



An iterative structure keeps on executing a certain piece of code until a certain condition is satisfied.

Basic Control Structures (BCSs)

Category	BCS	Structure	W_i
Sequence	Sequence (SEQ)		1
Branch	If-then-[else] (ITE)		2
	Case (CASE)		3

Category	BCS	Structure	W_i
Embedded component	Function call (FC)		2
	Recursion (REC)		3

Category	BCS	Structure	W_i
Iteration	For - do (R_i)		3
	Do - while (R_1)		3
	While-do (R_0)		3

Total Cognitive Weight

- The **total cognitive weight** of a software component, Wc, is defined as the sum of the cognitive weights of its q linear blocks composed of individual BCSs.
- Since each block may consist of *m* layers of nesting of BCSs, and each layer of *n* linear BCSs, Wc is calculated as follows:

$$W_c = \sum_{j=1}^{q} \left[\prod_{k=1}^{m} \sum_{i=1}^{n} W_c\,(j,k,i) \right] \qquad (1)$$

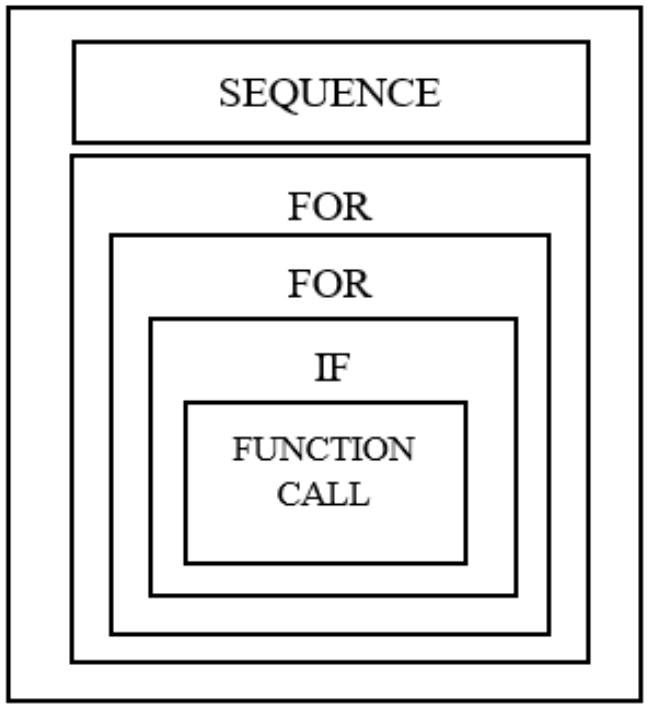
Total Cognitive Weight

- If there is no embedded BCS in any of the q blocks, i.e., m=1, then (1) can be simplified as follows:

$$W_c = \sum_{j=1}^{q} \sum_{i=1}^{n} W_c(j,i) \qquad (2)$$

Calculating Total Cognitive Weight

Source Code	Structure of BCSs	Cognitive Weight

public void bubbleSort(){
int out, in;
for(out=nElems-1; out>1; out--)
for(in=0; in<out; in++)
if(a[in] > a[in+1])
swap(in, in+1);
}
		$W_C = \sum_{j=1}^{q} [\prod_{k=1}^{m} \sum_{i=1}^{n} W_c\,(j,k,i)]$
		$W_C = 1 + 3\,(\,3\,(\,2\,(\,2\,)\,)\,)$
		$W_C = 1 + 36$
		$W_C = 37$

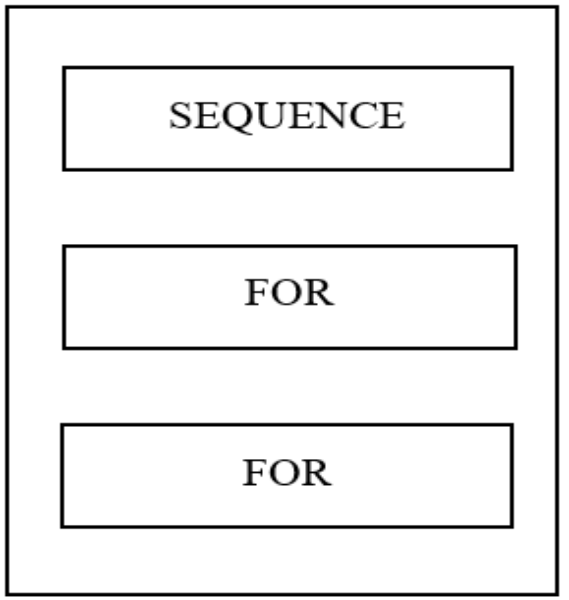
Calculating Cognitive Weight – Question

- Calculate the cognitive weight of following code segment:

```
public static void main(String[] args) {

  String[] modules = {"SEPQM", "DS", "ESD", "AF", "SA"};

  for (int i = 0; i < modules.length; i++) {
        System.out.println(modules[i]);
  }

  System.out.println("In reverse order:");

  for (int i = modules.length - 1; i >= 0; i--) {
        System.out.println(modules[i]);
  }
}
```

Calculating Cognitive Weight – Answer

Source Code	Structure of BCSs	Cognitive Weight

public static void main(String[] args) {
String[] modules = {"SEPQM", "DS", "ESD", "AF", "SA"};
for (int i = 0; i < modules.length; i++) {
System.out.println(modules[i]);
}
System.out.println("In reverse order:");
for (int i = modules.length - 1; i >= 0; i--) {
System.out.println(modules[i]);
}
}
		$W_C = \sum_{j=1}^{q} \sum_{i=1}^{n} W_c(j,i)$
		$W_C = 1 + 3 + 3$
		$W_C = 7$

Cognitive Complexity of a Basic Component

- The cognitive functional size of a basic software component that only consist of **one method**, S_f, is defined as a product of the sum of inputs and outputs ($N_{i/o}$) and the total cognitive weight (Wc). More formally, it can be defined as follows:

$$S_f = (N_i + N_o) \times W_c \qquad (3)$$

Cognitive Complexity of a Complex Component

- Based on (3), the cognitive functional size of a complex software component with ***n*** **methods**, $S_f(c)$, is defined as follows:

$$S_f(c) = \sum_{c=1}^{n} S_f(c) \qquad (4)$$

Cognitive Complexity of a Component-Based Software System

- The cognitive functional size of a component-based software system, $\hat{S}$, with ***p* components**, $\hat{S}_f$, is defined as follows:

$$\hat{S}_f = \sum_{k=1}^{p} S_f(k) \qquad (5)$$

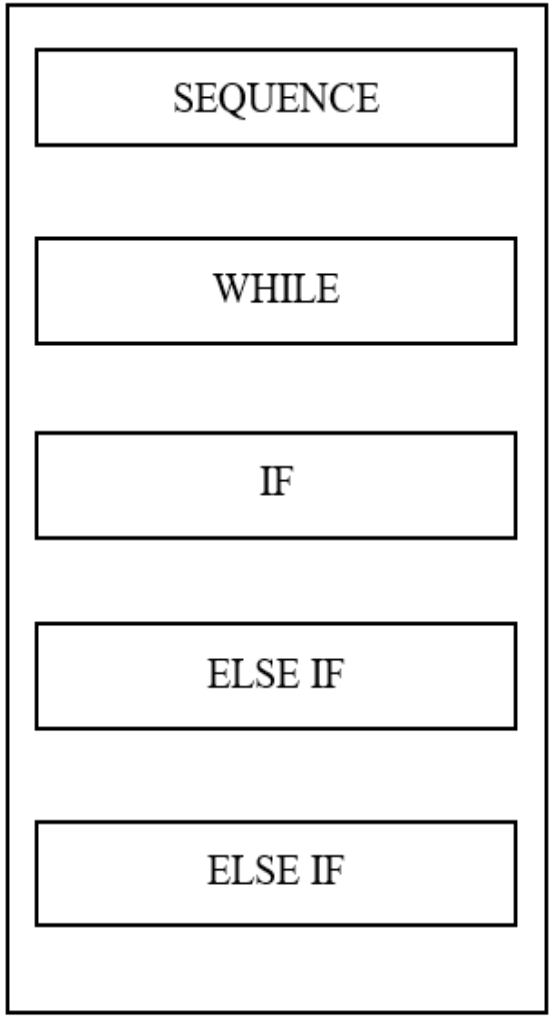
Calculating Cognitive Functional Size Value – Question

- Calculate the cognitive functional size of the following code segment:

```
import java.util.Scanner;
public class Results{
  public static void main(String[] args) {
    System.out.print("Enter your marks: ");
    Scanner sc = new Scanner(System.in);
    int marks = sc.nextInt();
    while(marks < 0 || marks > 100){
      System.out.print("Enter a valid mark: ");
      marks = sc.nextInt();
    }
    if (marks>75)
      System.out.println("A Pass");
    else if (marks<=75 && marks>65)
      System.out.println("B Pass");
    else if (marks<=65 && marks>45)
      System.out.println("C Pass");
    else
      System.out.println("Fail");
  }
}
```

Calculating Cognitive Functional Size Value – Answer

Source Code	Structure of BCSs	Cognitive Functional Size

import java.util.Scanner;
public class Results{
public static void main(String[] args) {
System.out.print("Enter your marks: ");
Scanner sc = new Scanner(System.in);
int marks = sc.nextInt();
while(marks < 0 || marks > 100){
System.out.print("Enter a valid mark: ");
marks = sc.nextInt();
}
if (marks>75)
System.out.println("A Pass");
else if (marks<=75 && marks>65)
System.out.println("B Pass");
else if (marks<=65 && marks>45)
System.out.println("C Pass");
else
System.out.println("Fail");
}
}
		$W_C = \sum_{j=1}^{q} [\prod_{k=1}^{m} \sum_{i=1}^{n} W_c(j,k,i)]$
		$W_C = 1 + 3 + 2 + 2 + 2$
		$W_C = 10$
		
		$N_i = 2$
		$N_o = 1$ (Only one S.O.P statement is excuted at a given time)
		
		$S_f = (N_i + N_o) \times W_c$
		$S_f = (2 + 1) \times 10$
		$S_f = 30$ [CWU]