I. Examples of Sequence Definitions

A. Iterative Sequence Definition of $a_0,\ a_1,\ a_2, ...,\ a_n$

1. Example 1:

   a. Explicit: $a_0,\ a_1,\ a_2, ...,\ a_n = 0,\ \frac{1}{2},\ \frac{2}{3},\ \frac{3}{4},\ \frac{4}{5},\ ...$

   b. Formula: $a_k = \frac{k}{k+1}$

2. Example 2:

   a. Explicit: $b_0,\ b_1,\ b_2, ...,\ b_n = -1,\ \frac{1}{4},\ \frac{-1}{9},\ \frac{1}{16},\ ...$

   b. Formula: $b_i = \frac{(-1)^{i+1}}{(i+1)^2}$

B. **Recursive Defintion of Summation:** $\boldsymbol{\sum_{k=0}^{N} A_k}$

1. $\sum_{k=0}^{N} A_k$ is the symbolic representation of the sum of N values, each individual value represented by A_k.

2. Definition:

   a. Basis Step: $N = 0$ $\quad \sum_{k=0}^{0} A_k = A_0$

   The sum of only the first value A_0 is that first value.

   b. Recursive or Inductive Step:

   $$\sum_{k=0}^{N+1} A_k = \left(\sum_{k=0}^{N} A_k\right) + A_{N+1}$$

   The sum of $N+1$ values is represented as the sum of the first N values $\left(\sum_{k=0}^{N} A_k\right)$ and the $(N+1)$st value, or A_{N+1}.

C. The Fibonacci Sequence:

1. Basis Step: $F_0 = 1 \qquad F_1 = 1$

2. Inductive Step: $F_k = F_{k-1} + F_{k-2}$

II. Introduction to Recursive Logic

A. A recursively defined function is a function defined in terms of itself.

B. There must be:

1. A Basis Step (equivalent to the basis step in the Theory of Mathematical Induction):

   The value of the function for the first value(s) for which the function is defined must be explicitly stated.

2. A rule for determining the value of the function at some arbitrary integer N using the values computed for smaller integers.

C. The process of ***recursion*** involves breaking a problem into subproblems, often proceeding through many levels of subproblems.

1. When the lowest level subproblem is solved, each solution is consolidated into the solution of the main problem.

2. As a result, a recursively defined function is a function defined in terms of itself.

3. The definition must have an explicit statement for the first (or last) value(s) returned by the function.

4. All other values are expressed in terms of the solutions to subproblems.

II. Recursive or Inductive Function Definitions

A. Two Steps are Required (as in Mathematical Induction)

B. A Basis Step: The value of the function for the first integer being processed (0?)must be explicitly specified.

C. Recursive (or Inductive) Step:
The rule or logic for determining the value of the function at some integer N using its values for integers $I < N$

E. Recursive Sequence Definition: **Recurrence Relation**

1. A ***recurrence relation*** for a sequence $a_0, a_1, a_2, ..., a_n$ is a formula that relates each term a_k to certain of its predecessors $a_{k-1}, a_{k-2}, .., a_{k-i}.$, where i is an integer such that $k - i \geq 0$

2. The initial conditions for such a recurrence relation specify the values of $a_0, a_1, ..., a_{i-1}$ if i is a fixed integer or $a_0, a_1, ..., a_m$, where m is an integer with $m \geq 0$, if i depends upon k.

III. Example 1: A Recursive Version of Euclid's Algorithm for finding the GCD (greatest common divisor) of A and B:

A. Problem: Given two integers, determine the largest integer that will divide both of them with no remainder.

B. Recursive Algorithm:

1. Let $A = B \times Q + R$ where A, B, Q, and R are integers.

2. Then the greatest common divisor of A and B is the greatest common divisor of B and R.

C. Computer Algorithm Consideration:

1. If either $A = 0$ or $B = 0$ is 0 return an error value.

2. If $A < B$ interchange the values of A and B so that $A > B$

D. Basis Step:

1. Compute the remainder of $\frac{A}{B}$, i.e., the modulus of A and B or $M = A\%B$

2. if $M = 0$, B is the greatest common divisor of A and B

E. Inductive Step: Compute the greatest common divisor of B and M

1. Invoke the recursive function with $A = B$ and $B = M$.

2. Compute the the modulus of A and B or $M = A\%B$

3. if $M = 0$, B is the greatest common divisor of A and B

4. If $M \neq 0$ repeat the inductive step.

F. The Logic stated in Java

```
public static int gcdCalc(int   X,  int Y)
                {
                int First;
                int Scnd;

                if (X >= Y) {   First = X;  Scnd = Y;   }
                else          {   First = Y;  Scnd = X;   }

                if (Scnd == 0) return 0;
                else    {
                        int M = First % Scnd;

                        if (M == 0) return Scnd;

                        else return gcdCalc(Scnd, M);
                        }
                }
```

G. Example Output: (with a trace display)

```
run:
Enter the first integer to be processed: 48
Enter the second integer to be processed: 26

Invoking rcrsEuclidGCD with A = 48 and B = 26

Invoking rcrsEuclidGCD with A = 26 and B = 22

Invoking rcrsEuclidGCD with A = 22 and B = 4

Invoking rcrsEuclidGCD with A = 4 and B = 2

The greatest common denominator of 48 and 26 is 2
BUILD SUCCESSFUL (total time: 15 seconds)
```

IV. Example 2: The Fibonacci Sequence

A. History: Problem proposed in 1602 by Leonardo of Pisa (1175 - 1250), commonly known as Fibonacci.

B. Problem Statement:

1. Initial situation: A single pair (male and female) of rabbits is born at the beginning of a year.
2. Conditions:
   a. Rabbit pairs are not fertile for the first month of their life.
   b. After the first month the pair gives birth to one new male/female pair each month.
   c. No rabbits die.
3. Question: After *N* months how many rabbit pairs are in existence?

C. Sequential Solution:

1. In month 0 we have one rabbit pair.

2. In month 1 we have one rabbit pair because the initial pair cannot reproduce until they are one month old.

3. In month 2 we have one reproducing pair which has produced one immature pair, or 2 pairs.

4. In month 3 we have one reproducing pair that has reproduced once again and the immature pair of the previous month, or 3 pairs.

5. In month 4 we have two reproducing pairs (the original and the oldest pair of their offspring) that each generate an immature pair, or 5 pairs of rabbits.

6. The sequence being generated is:
$F_0,\ F_1,\ F_2,\ \ldots = 1,\ 1,\ 2,\ 3,\ 5,\ 8,\ \ldots$

D. Recursive Definition:

1. Basis Step: $F_0 = 1 \qquad F_1 = 1$

2. Inductive Step: $F_k = F_{k-1} + F_{k-2}$

V. Recursive Logic Necessary to Compute F_K

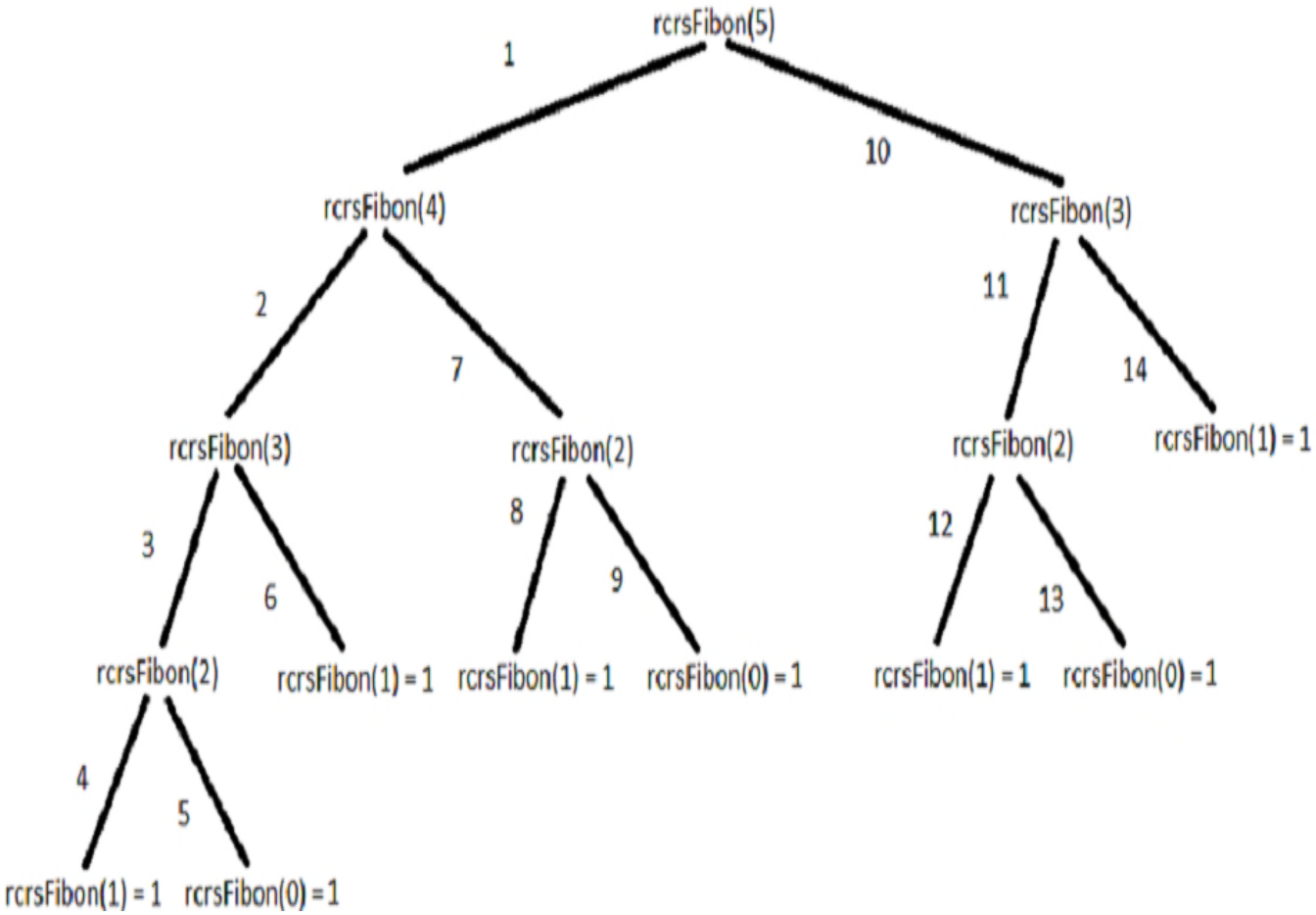
A. Java Code

```
public static long rcrsFibon(int N)
    {
    if (N == 0) return 1;

    else if (N == 1) return 1;

    else return rcrsFibon(N - 1) + rcrsFibon(N - 2);
    }
```

B. The Recursive Logic Flow Necessary to Compute an Element of the Fibonacci Sequence is shown below for the fifth Fibonacci number:



VI. Iterative Logic Necessary to Compute F_K

A. Algorithm

1. Compute the remainder $M = \text{A\%B}$

2. If $M = 0$ stop.

3. Otherwise:
   a. $A = B$

   b. $B = M$

   c. Repeat steps 1 - 3.

B. Java Code:

```
public static long iterFib(int N)
    {
    long   Fib0   =   1;
    long   Fib1   =   1;

    if (N == 0) return Fib0;
    else if (N == 1) return Fib1;
    else   {
        long   F0  =   Fib0;
        long   F1  =   Fib1;
        long   fSum   =   0;
        for (int i = 1; i <= N; i++)
            {
            fSum = F1 + F0;    F0 = F1;
            F1 = fSum;
            }

        return fSum;
        }
    }
```

VII. Explicit Statement of F_K

A. Explicit Solution Derivation Logic:

1. Recurrence Relation:

   a. Basis Step: $F_0 = 1 \qquad F_1 = 1$

   b. Inductive Step: $F_k = F_{k-1} + F_{k-2}$

   c. Hence: The Fibonacci sequence is defined by a linear, homogeneous, second-order recurrence relation.

2. For $k = 2$ we have: $F_2 - F_1 - F_0 = 0$

3. We associate $F_2 - F_1 - F_0 = 0$

   with $t^2 - t - 1 = 0$

4. Solutions: $t_1 = \frac{1+\sqrt{5}}{2}$ and $t_2 = \frac{1-\sqrt{5}}{2}$

5. The value $\frac{1+\sqrt{5}}{2}$ is known as the ***Golden ratio,*** or ϕ and the value $\frac{1-\sqrt{5}}{2}$ is it's conjugate, or ϕ^t

6. Then our solution is of the form:

$$F_N = B\left(\frac{1+\sqrt{5}}{2}\right)^N + D\left(\frac{1-\sqrt{5}}{2}\right)^N$$

or

$$F_N = B\phi^N + D{\phi^t}^N$$

5. Initial Conditions:

$$B\phi + D\phi^t = 1 = F_1$$
$$B\phi^2 + D{\phi^t}^2 = 2 = F_2$$

$$\overline{\qquad\qquad\qquad\qquad\qquad}$$

$$B\phi^2 + D\phi\phi^t = \phi$$
$$B\phi^2 + D{\phi^t}^2 = 2$$

$$\overline{\qquad\qquad\qquad\qquad\qquad}$$

$$D\phi^t(\phi - \phi^t) = -2$$

Therefore: $D = \frac{-2}{\phi^t(\phi - \phi^t)} = \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)$

6. Substituting for D gives us: $B = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)$

7. Therefore, the Explicit Formula is:

$$F_N = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^{N+1} - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^{N+1}$$

or: $F_N = \frac{1}{\sqrt{5}}\phi^{N+1} - \frac{1}{\sqrt{5}}{\phi^t}^{N+1}$

B. Verification Using Strong Mathematical Induction:

1. Basis Steps:

a. $F_1 = \frac{1}{\sqrt{5}}\phi^{1+1} - \frac{1}{\sqrt{5}}\phi^{t\,1+1} = \frac{1}{\sqrt{5}}\phi^2 - \frac{1}{\sqrt{5}}\phi^{t\,2}$

$= \frac{1}{\sqrt{5}}\left(\frac{1+2\sqrt{5}+5-1+2\sqrt{5}-5}{4}\right) = \frac{1}{\sqrt{5}} \times \frac{4\sqrt{5}}{4}$

$= 1$

b. $F_2 = \frac{1}{\sqrt{5}}\phi^{2+1} - \frac{1}{\sqrt{5}}\phi^{t\,2+1} = \frac{1}{\sqrt{5}}\phi^3 - \frac{1}{\sqrt{5}}\phi^{t\,3}$

$= \frac{1}{\sqrt{5}}\left(\frac{1+8\sqrt{5}+15-1+8\sqrt{5}-15}{8}\right) = \frac{1}{\sqrt{5}} \times \frac{8\sqrt{5}}{8}$

$= 1$

2. Inductive Assumption:
$F_K = \frac{1}{\sqrt{5}}\phi^{K+1} - \frac{1}{\sqrt{5}}\phi^{t\,K+1}$
is the Kth element of the Fibonacci sequence for $1 \le K \le N$

3. Inductive Step: Consider $F_{K+1} = F_K + F_{K-1}$

4. $F_K + F_{K-1} = \frac{1}{\sqrt{5}}\phi^{K+1} - \frac{1}{\sqrt{5}}\phi^{t\,K+1} + \frac{1}{\sqrt{5}}\phi^K - \frac{1}{\sqrt{5}}\phi^{t\,K}$

$= \frac{\phi^{K+1} - \phi^{t\,K+1} + \phi^K - \phi^{t\,K}}{\sqrt{5}} = \frac{\phi^K(\phi+1) - \phi^{t\,K}(\phi^t+1)}{\sqrt{5}}$

according to our inductive assumption.

5. We note that: $\phi^2 = \left(\frac{1+\sqrt{5}}{2}\right)^2 = \frac{1+2\sqrt{5}+5}{2}$

$= \frac{6+2\sqrt{5}}{4} = \frac{3+\sqrt{5}}{2}$

$= \frac{2}{2} + \frac{1+\sqrt{5}}{2}$

$= 1 + \phi$

and: $\phi^{t2} = \left(\frac{1-\sqrt{5}}{2}\right)^2 = \frac{1-2\sqrt{5}+5}{2}$

$= \frac{6-2\sqrt{5}}{4} = \frac{3-\sqrt{5}}{2}$

$= \frac{2}{2} + \frac{1-\sqrt{5}}{2}$

$= 1 + \phi^t$

6. Then: $F_{K+1} = F_K + F_{K-1} = \frac{\phi^K(\phi+1) - \phi^{tK}(\phi^t+1)}{\sqrt{5}}$

$= \frac{\phi^K\phi^2 - \phi^{tK}\phi^{t2}}{\sqrt{5}} = \frac{\phi^{(K+1)+1} - \phi^{t(K+1)+1}}{\sqrt{5}}$

$= \frac{1}{\sqrt{5}}\phi^{(K+1)+1} - \frac{1}{\sqrt{5}}\phi^{t(K+1)+1}$

$= F_{K+1}$

7. Therefore, as proven by Strong Induction,

$$F_K = \frac{1}{\sqrt{5}}\phi^{K+1} - \frac{1}{\sqrt{5}}\phi^{tK+1}$$

is the Kth element of the Fibonacci sequence

C. Java Code:

```
public static double explicitFibon(int N)
        {
        double  sqRootFive   =   Math.sqrt(5.0);
        double  One =   1.0;
        double  Two =   2.0;
        double  Factor  =   One / sqRootFive;

        double  Term1   =   (One + sqRootFive)/Two;
        double  Term2   =   (One - sqRootFive)/Two;
        double  Power   =   (double) N + One;

        double  Result  =   Factor*Math.pow(Term1, Power)
                            - Factor*Math.pow(Term2, Power);

        return Result;
        }
```

VIII. Comparison of Results

A. Recursive Solution: The 12th Fibonacci number is 233

B. Iterative Solution: The 12th Fibonacci number is 233

C. Explicit Solution: The 12 th Fibonacci number is 233

IX. Example 3: Ackermann's Function

A. Definition:

1. $A(M,\ N) = 2 \times N \qquad if \qquad M = 0$

2. $A(M,\ N) = 0 \qquad if \qquad M \geq 1\, and\ N = 0$

3. $A(M,\ N) = 2 \qquad if \qquad M \geq 1\, and\ N = 1$

4. $A(M,\ N) = A(M-1,\ A(M,\ N-1))$
   $if\ M \geq 1\, and\, N \geq 2$

B. Recursive Java Code

```
public static long AckerMann(long M, long N)
    {
    if (M == 0) return 2*N;

    else if (M >= 1 && N == 0) return 0;

    else if (M >= 1 && N == 1) return 2;

    else if (M >= 1 && N >= 2)
        return AckerMann(M - 1, AckerMann(M, N - 1));

    else return -5;       /*  A value signifying an error.  */
    }
```

3. Output of Recursive Implementation:

```
run:
Enter a value for M: 2
Enter a value for N: 1
Ackermann(2, 1) = 2
BUILD SUCCESSFUL (total time: 6 seconds)

run:
Enter a value for M: 2
Enter a value for N: 4
Ackermann(2, 4) = 65536
BUILD SUCCESSFUL (total time: 6 seconds)

run:
Enter a value for M: 3
Enter a value for N: 3
Ackermann(3, 3) = 65536
BUILD SUCCESSFUL (total time: 4 seconds)
```

C. Stack Space: Machine Resource Limitation Affecting Recursion

1. Each invocation of a sub-routine or function requies an entry to be made onto the system stack.

2. Each entry requires stack memory space.

3. The trace shown below indicates the number of invocations necessary to compute A(2, 3)

```
run:
Enter a value for M: 2
Enter a value for N: 3
Invoking Ackermann(2, 3)
Invoking Ackermann(2, 2)
Invoking Ackermann(2, 1)
Invoking Ackermann(1, 2)
Invoking Ackermann(1, 1)
Invoking Ackermann(0, 2)
Invoking Ackermann(1, 4)
Invoking Ackermann(1, 3)
Invoking Ackermann(1, 2)
Invoking Ackermann(1, 1)
Invoking Ackermann(0, 2)
Invoking Ackermann(0, 4)
Invoking Ackermann(0, 8)
Invoking Ackermann(2, 3)
Invoking Ackermann(2, 2)
Invoking Ackermann(2, 1)
Invoking Ackermann(1, 2)
Invoking Ackermann(1, 1)
Invoking Ackermann(0, 2)
Invoking Ackermann(1, 4)
Invoking Ackermann(1, 3)
Invoking Ackermann(1, 2)
Invoking Ackermann(1, 1)
Invoking Ackermann(0, 2)
Invoking Ackermann(0, 4)
Invoking Ackermann(0, 8)
Ackermann(2, 3) = 16
BUILD SUCCESSFUL (total time: 11 seconds)
```

4. The limit of available memory for the stack can be reached quite quickly relative to parameter size, as shown below:

```
run:
Enter a value for M: 4
Enter a value for N: 3
Exception in thread "main" java.lang.StackOverflowError
```