Part1:

1)Host-parasite Interaction: This recurrence relation shows the interaction between two populations that live together in discrete time. It assumes that parasites search for hosts randomly and assumed that both parties are distributed as non-contigiously. Adding denisty dependence allows stable coexistance.

N_{t+1} = \lambda N_t e^{-aP_t} \,

P_{t+1} = N_t(1-e^{-aP_t}), \,

Nt = hosts at time t

Pt = parasites at time t

ʎ = reproductive rate of host

= Probablity that the host will survive Pt parasites

Towers of Hanoi: A puzzle game where 3 certain rules must be obeyed. Only one disk can be moved at a time. A disk can only be moved if it is at the top of a stack. A bigger disk can not be placed on a smaller disk. This puzzle can be solved using three steps: 1) Moving n-1 disks from first post to third post (Done in Tn-1 steps). 2) Moving the last disk to the second post. 3) Moving n-1 disks from third post to second post ( Done in Tn-1 steps.) This algorithm results in the following recurrence relation:

Hn=2Hn−1+1

H1= 1

N = number of disks

Hn=number of moves required to finish this puzzle that has N disks.

Compound Interest: Addition of interest to an already existing currency is called compound interest. At the end of each period the resulting value is calculated by altering the value from the previous period which results in the following recurrence relation:

P(t) = i.P(t-1)

t = period

i =interest rate

P = principal sum

Fibonnaci Sequence: Used for finding the golden ratio, shallow diagonals in pascal triangle. It is also used as a model for the growth of rabbit populations.

F_n = F_{n-1}+F_{n-2}

Where F0=0 F1=1

Lucas Numbers: Same as Fibonacci numbers. But for L0 = 2 and L1=1. Every ratio of

Padovan Sequence: This sequence is related to the sums of binomial coefficients.

 P(n)=P(n-2)+P(n-3) 

With initial conditions:

P(0)=P(1)=P(2)=1

Part 2:

1)i) Xn=(97/100)\*xn-1+65+360

x0 = 5000

x1= (97/100)\*x0+65+180

ii)

MATLAB code:

function [] = reservoirRecursive( x )

count = 0;

fprintf('x%d = %4.2f \n',count,x)

count = count + 1;

x = (97/100)\*x+245;

fprintf('x%d = %4.2f \n',count,x)

while x < 8166.66

count = count + 1;

y = x\*(97/100)+(65+180);

fprintf('x%d = %4.2f \n',count,y)

x = y;

end

end

reservoirRecurrence(5000)

iii) Stable. The value of xn converges to 8166.66 at n = 430.

2)i) Xn=xn-1\*(109/100)+k\*1000+(k-1)\*2500

x0 = 40000

k = 1 if n % 2 = 1

k = 0 if n % 2 = 0

ii)

|  |  |
| --- | --- |
| n | x(n) |
| 1 | 44600.00 |
| 2 | 46114.00 |
| 3 | 51264.26 |
| 4 | 53378.04 |
| 5 | 59182.07 |

MATLAB code:

function [] = bankAccountRecursive( x )

count = 0;

fprintf('x%d = %4.2f \n',count,x)

while x > 0

count = count + 1;

if mod(count,2) == 1

y = x\*(109/100)+1000;

fprintf('x%d = %4.2f \n',count,y)

x = y;

end

if mod(count,2) == 0

y = x\*(109/100)-2500;

fprintf('x%d = %4.2f \n',count,y)

x = y;

end

end

end

bankAccountRecurrence(40000)

iii) Unstable. The value of xn converges to infinity.