**DESIGN ASPECT**

Let's define Fm , Im , Ic and Fc, to represent the state of the river at any given time: Fm missionaries and Fc cannibals on the left, Im missionaries and Ic cannibals on the right.

The problem starts out in the state where all 3 Missionaries and 3 Cannibal are on right side which means that Im and Ic are both 3 and Fm and F are both zero, and we want them to get on the left side of river without fail conditions that a boat can carry at most 2 men, if there are missionaries present on the bank or the boat, they cannot be outnumbered by cannibals (if they were, the cannibals would eat the missionaries). The boat cannot cross the river by itself with no people on board. and BO( , )AT is written on left if the boat is on the left and BO( , )AT is on right if the boat is on the right.

**Base Case :** The base case includes the initial display in with the number of both missionaries and cannibal on the right side are 3 and that on the left side is 0 , and then after the recursive function is completed the base case contain the final display in which the number of both missionaries and cannibals on the left side becomes 3 and that on the right side becomes zero , hence for the base case initially Fm =0 , Fc =0, Im = 3, Ic =3 and it changes as missionaries and cannibal start moving.

**Algorithm :**

The Algorithm of this problem depends on the fact that you can have only cannibals on one side, and they won't be able to do anything to the missionaries no matter how many there are on one side. And the given condition that are give and are necessary to follow. So the program basis upon the follow procedure

{

If( one Missionary can move from right to left)

Cross( Im-1, Ic , Fm+1, Fc )

If( two Missionary can move from right to left)

Cross(Im-2, Ic , Fm+2, Fc )

If( one Missionary and one cannibal can move from right to left)

Cross(, Im-1, Ic-1 , Fm+1, Fc+1 )

Until the condition (Im=0, Ic=0 , Fm=3, Fc=3)}

Hence according to upper condition the program runs as follow:

For the case let as say M indicates Missionaries and C indicates Cannibals of M being more than C and (B) indicates that boat is on which side :

\*\**PROGRAM STARTS*\*\*

Initially the positions are : 0M , 0C and 3M , 3C (B)

Now let’s send 2 Cannibals to left of bank : 0M , 2C (B) and 3M , 1C

Send one cannibal from left to right : 0M , 1C and 3M , 2C (B)

Now send the 2 remaining Cannibals to left : 0M , 3C (B) and 3M , 0C  
Send 1 cannibal to the right : 0M , 2C and 3M , 1C (B)

Now send 2 missionaries to the left : 2M , 2C (B) and 1M . 1C

Send 1 missionary and 1 cannibal to right : 1M , 1C and 2M , 2C (B)

Send 2 missionaries to left : 3M , 1C (B) and 0M , 2C

Send 1 cannibal to right : 3M , 0C and 0M , 3C (B)

Send 2 cannibals to left : 3M , 2C (B) and 0M , 1C

Send 1 cannibal to right : 3M , 1C and 0M , 2C (B)’

Send 2 cannibals to left : 3M , 3C (B) and 0M , 0C

\*\**PROGRAM ENDs*\*\*

**Recursive function**

In the above program the function solution () is the recursive function which calls itself every time the final condition is false and it ends with the final condition is obtained.

**Output:**

In our program the output occur when our final required condition is achieved which is that when Im =0 ,Ic =0, Fm =3 ,Fc =3 it means when all 6 of the missionaries and cannibals have moved on the other side of river with the accomplishment of all given condition than the given output is showed

M,M,M,C,C,C (BO( , )AT)---Water

Hence we have reached our final required condition , so here the recursive function will stop this final display will be showed and the program finishes