CSC215

Math and Computer Science

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

A Legacy of Excellence

Fibonacci Sequence

- The Spirals on a Shell
- Petals on a flower (the number of petals)
- Pine cones
- Sunflowers
- Leaf arrangements on a tree

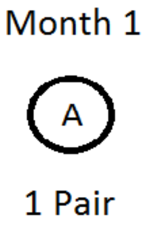
Fibonacci - Rabbits

Rules

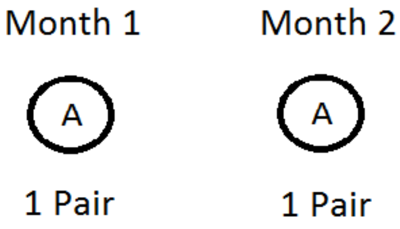
1. Rabbits never die
2. Rabbits will produce 1 pair after 2 months, In the 3rd month
3. They will always produce 1 pair (1 male and 1 female)

Any ideas on how to put this into a recursive function?

Month 1



Month 2



Month 1

A

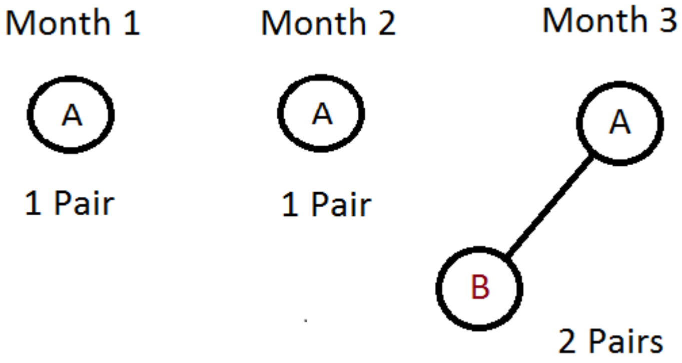
1 Pair

Month 2

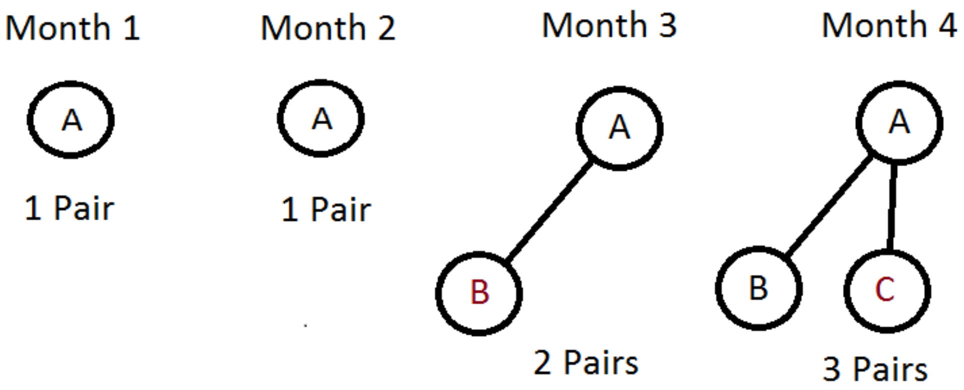
A

1 Pair

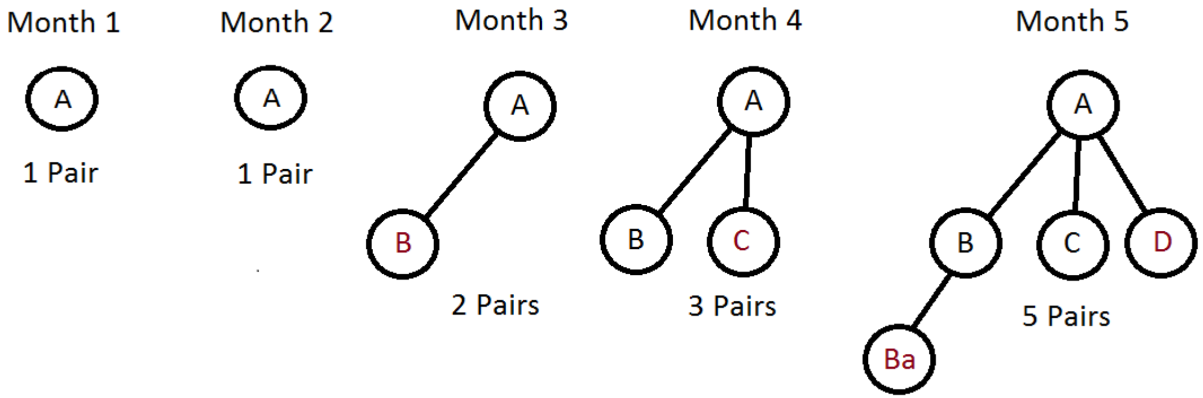
Month 3



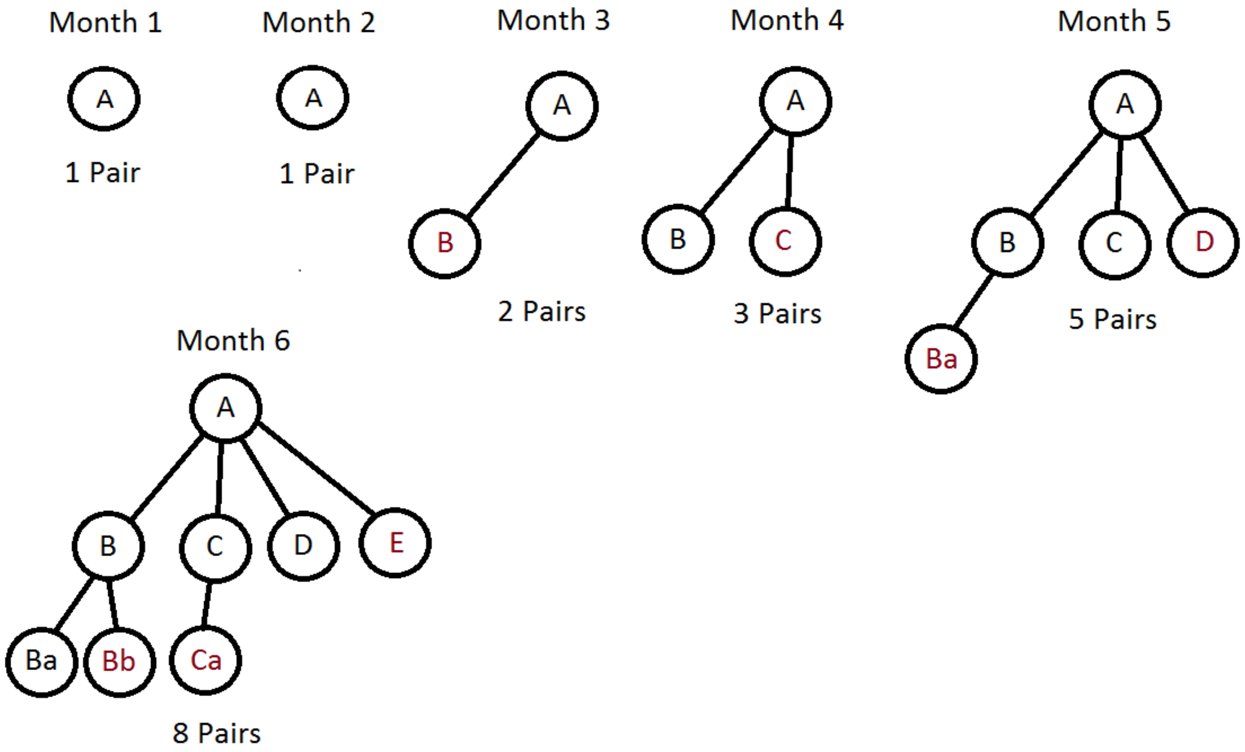
Month 4


Month 1
A
1 Pair
Month 2
A
1 Pair
Month 3
A
B
2 Pairs
Month 4
A
B
C
3 Pairs

Month 5

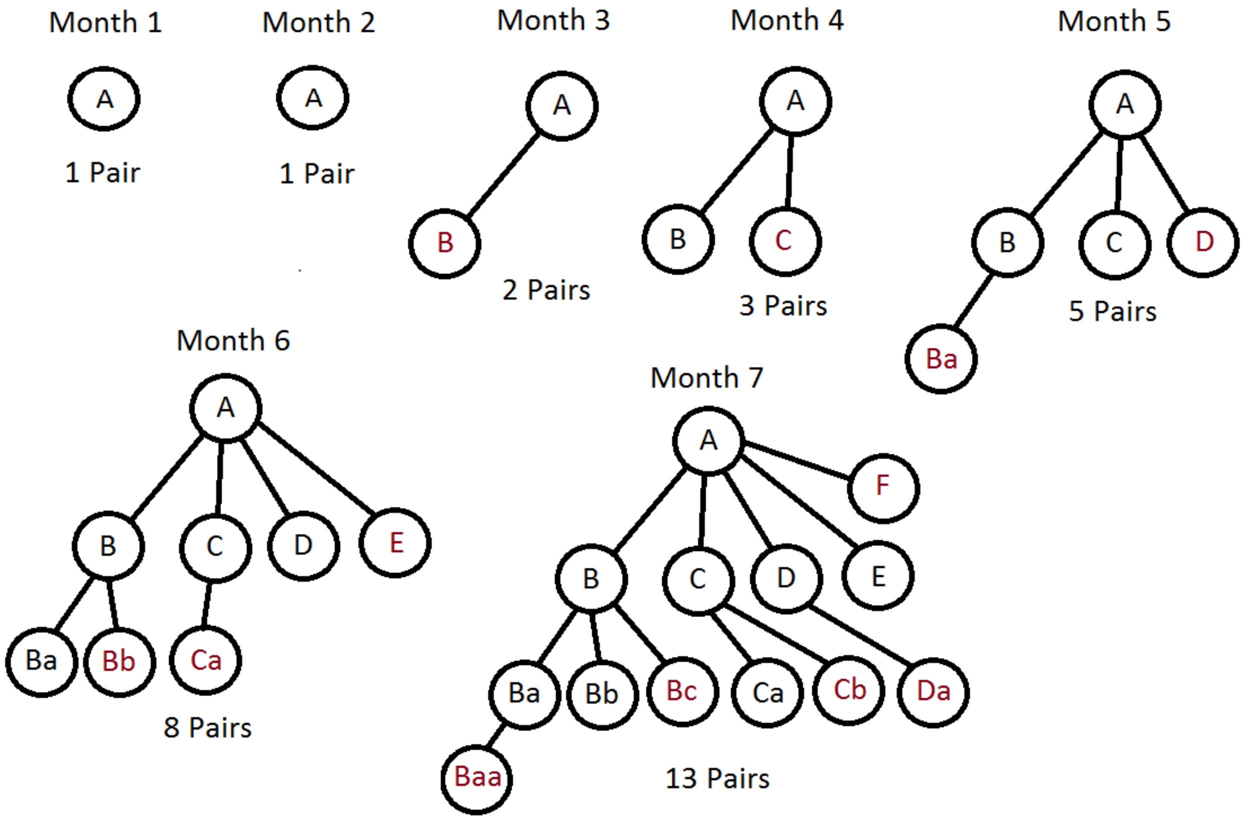

Month 1
A
1 Pair
Month 2
A
1 Pair
Month 3
A
B
2 Pairs
Month 4
A
B
C
3 Pairs
Month 5
A
B
C
D
Ba
5 Pairs

Month 6



SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Month 7



SOUTH DAKOTA M SCHOOL OF MINES & TECHNOLOGY

Month 8

- Sum of Month 7 + Month 6 = 13 Pairs + 8 Paris = 21 Pairs

Fibonacci Criteria

$n = 1$	1 Pair
$n = 2$	1 Pair
$n > 2$	Fibonacci(n – 1) + Fibonacci(n – 2)

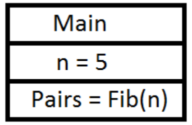
Writing the Code

```
unsigned long int fib(unsigned long int n)
{
    // write the base case first
    if( n <= 2 )
        return 1;



}
```

Writing the Code

```
unsigned long int fib(unsigned long int n)
{
    // write the base case first
    if( n <= 2 )
        return 1;

    // return the computed average
    return fib( n-1 ) + fib( n-2 );
}
```

Box Method

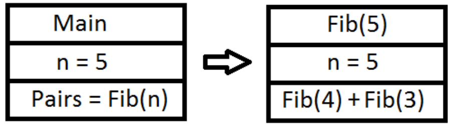


Main

n = 5

Pairs = Fib(n)

Box Method



Main

n = 5

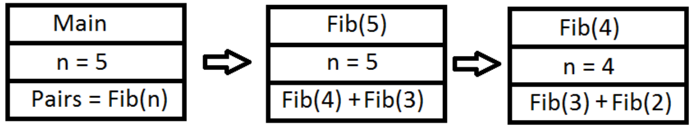
Pairs = Fib(n)

Fib(5)

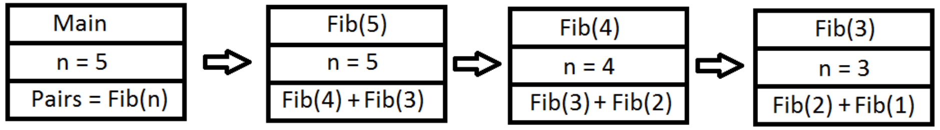
n = 5

Fib(4) + Fib(3)

Box Method



Box Method



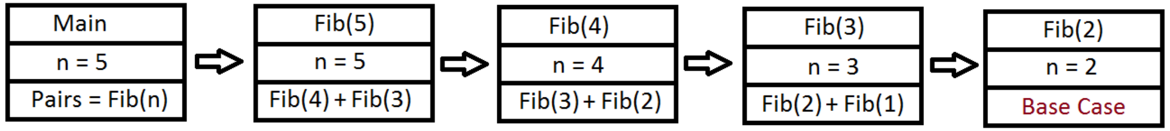
Main
n = 5
Pairs = Fib(n)

Fib(5)
n = 5
Fib(4) + Fib(3)

Fib(4)
n = 4
Fib(3) + Fib(2)

Fib(3)
n = 3
Fib(2) + Fib(1)

Box Method



Main
n = 5
Pairs = Fib(n)

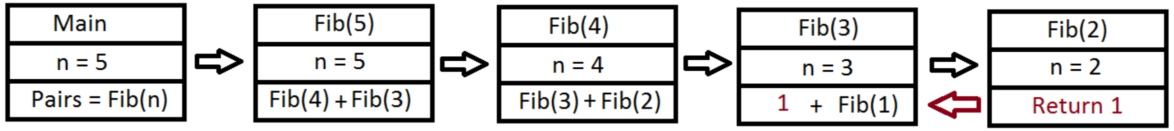
Fib(5)
n = 5
Fib(4) + Fib(3)

Fib(4)
n = 4
Fib(3) + Fib(2)

Fib(3)
n = 3
Fib(2) + Fib(1)

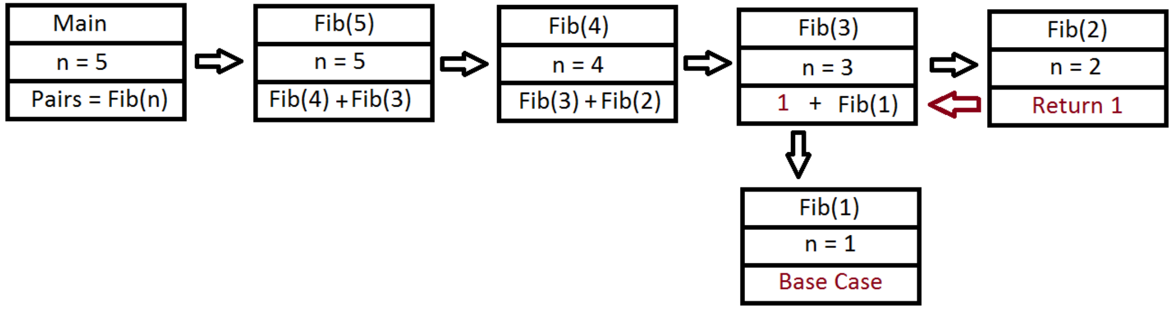
Fib(2)
n = 2
Base Case

Box Method

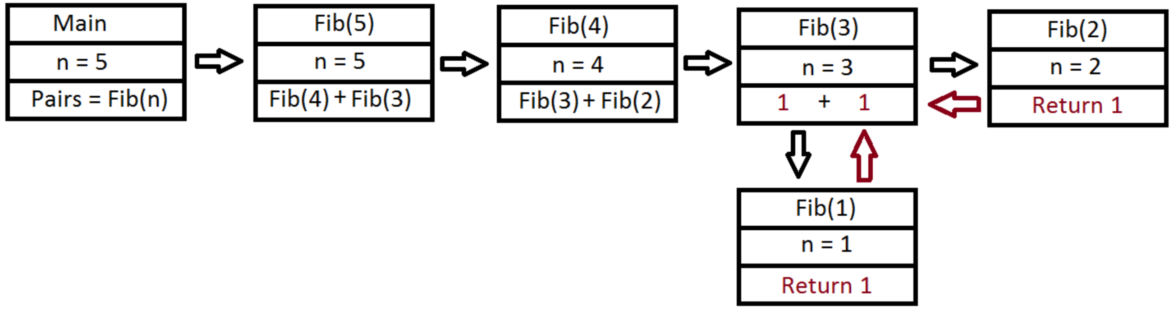


Main
n = 5
Pairs = Fib(n)
Fib(5)
n = 5
Fib(4) + Fib(3)
Fib(4)
n = 4
Fib(3) + Fib(2)
Fib(3)
n = 3
1 + Fib(1)
Fib(2)
n = 2
Return 1

Box Method

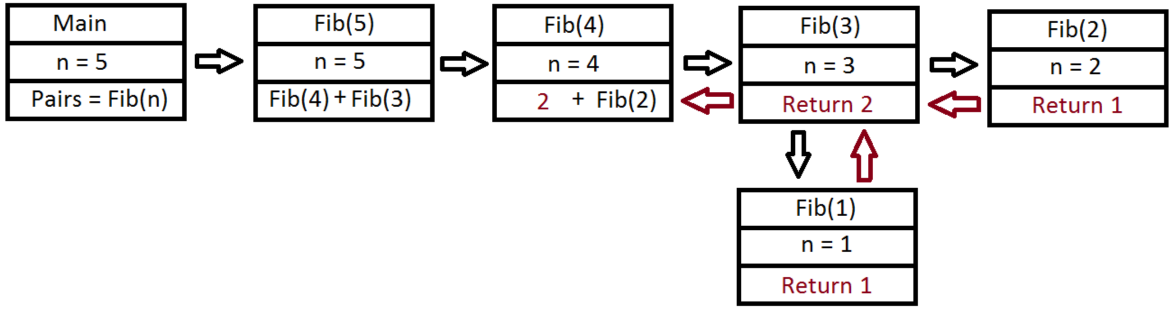

Main
n = 5
Pairs = Fib(n)
Fib(5)
n = 5
Fib(4) + Fib(3)
Fib(4)
n = 4
Fib(3) + Fib(2)
Fib(3)
n = 3
1 + Fib(1)
Fib(2)
n = 2
Return 1
Fib(1)
n = 1
Base Case

Box Method



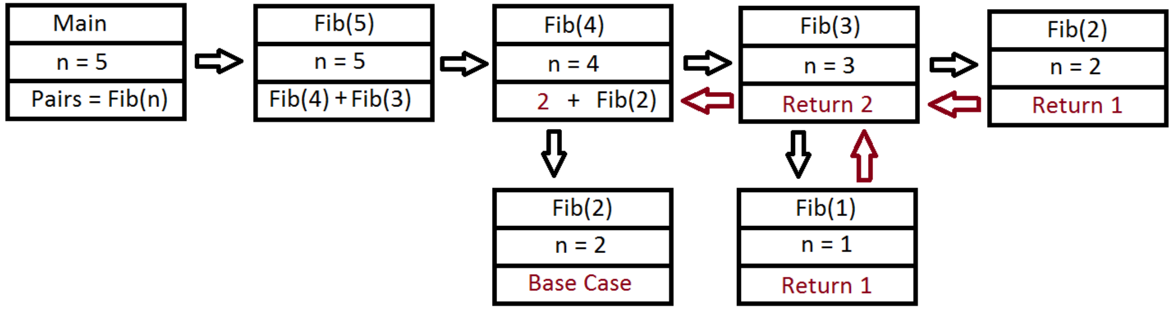
SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Box Method



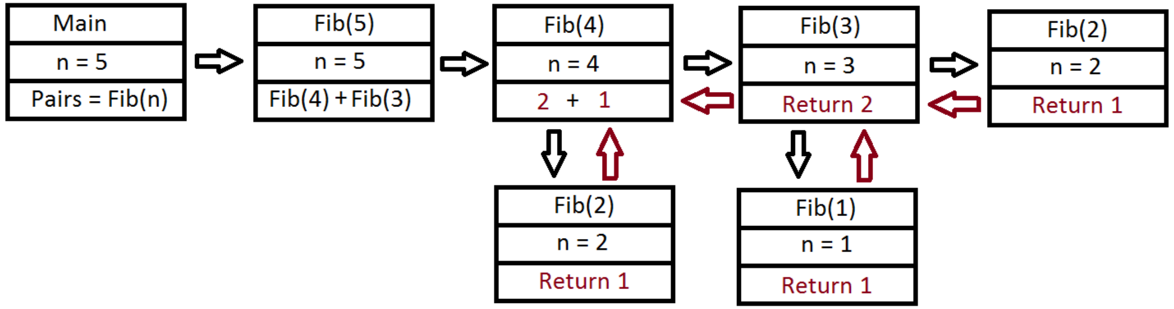
SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Box Method



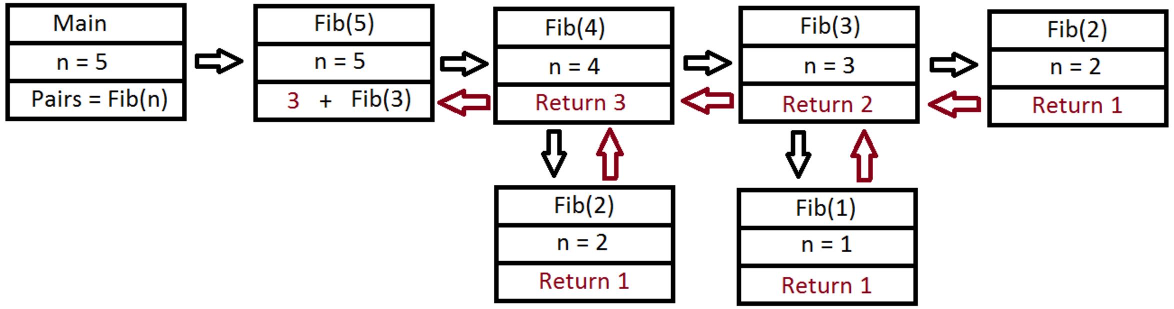
SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Box Method



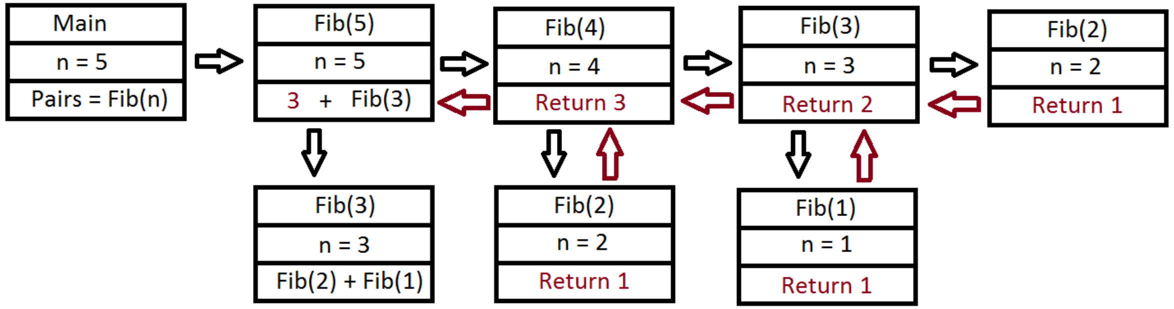
SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Box Method



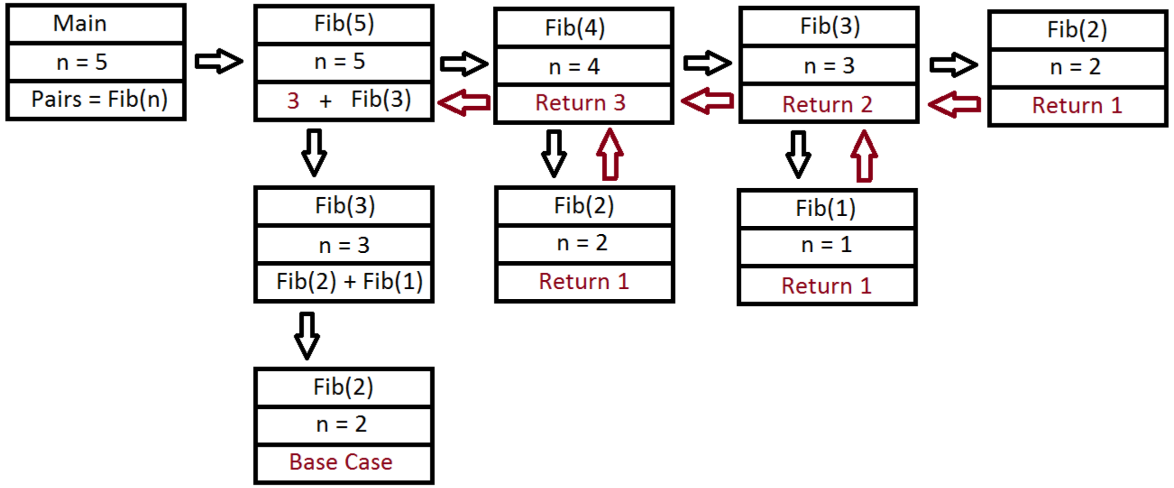
SOUTH DAKOTA M SCHOOL OF MINES & TECHNOLOGY

Box Method



SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Box Method



Main
n = 5
Pairs = Fib(n)

Fib(5)
n = 5
3 + Fib(3)

Fib(4)
n = 4
Return 3

Fib(3)
n = 3
Return 2

Fib(2)
n = 2
Return 1

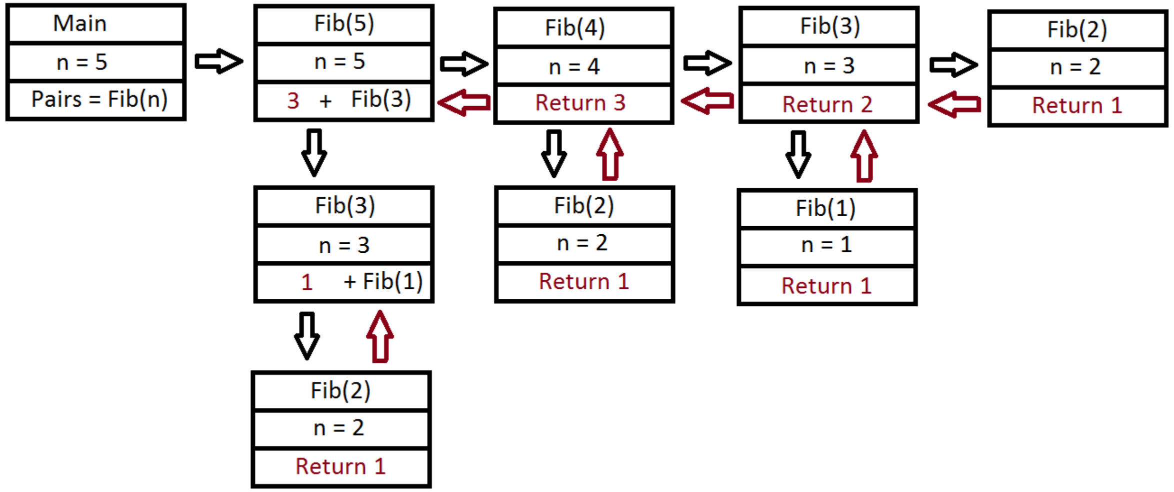
Fib(2)
n = 2
Return 1

Fib(1)
n = 1
Return 1

Fib(3)
n = 3
Fib(2) + Fib(1)

Fib(2)
n = 2
Base Case

Box Method



Main
n = 5
Pairs = Fib(n)

Fib(5)
n = 5
3 + Fib(3)

Fib(4)
n = 4
Return 3

Fib(3)
n = 3
Return 2

Fib(2)
n = 2
Return 1

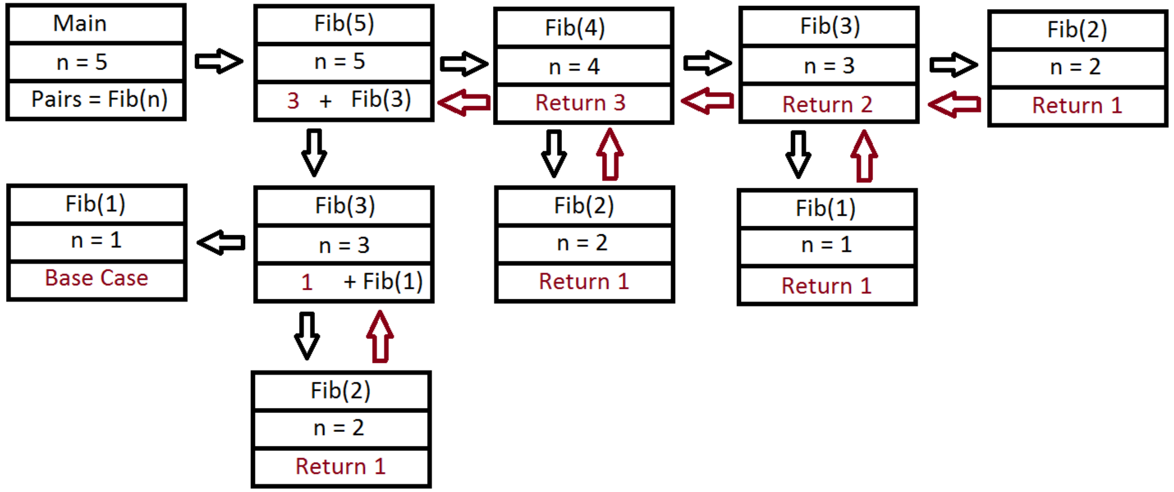
Fib(3)
n = 3
1 + Fib(1)

Fib(2)
n = 2
Return 1

Fib(1)
n = 1
Return 1

Fib(2)
n = 2
Return 1

Box Method



Main
n = 5
Pairs = Fib(n)

Fib(5)
n = 5
3 + Fib(3)

Fib(4)
n = 4
Return 3

Fib(3)
n = 3
Return 2

Fib(2)
n = 2
Return 1

Fib(1)
n = 1
Base Case

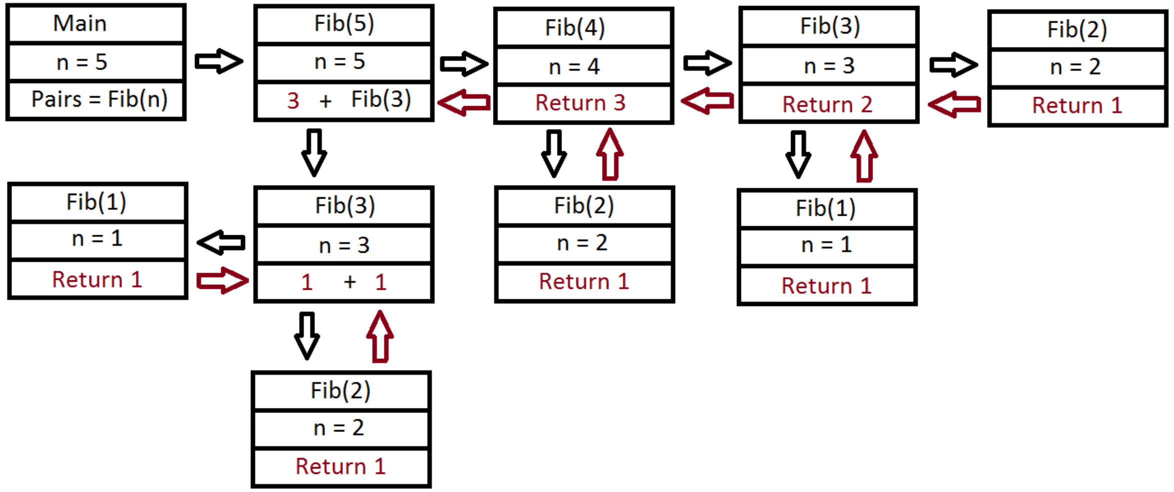
Fib(3)
n = 3
1 + Fib(1)

Fib(2)
n = 2
Return 1

Fib(1)
n = 1
Return 1

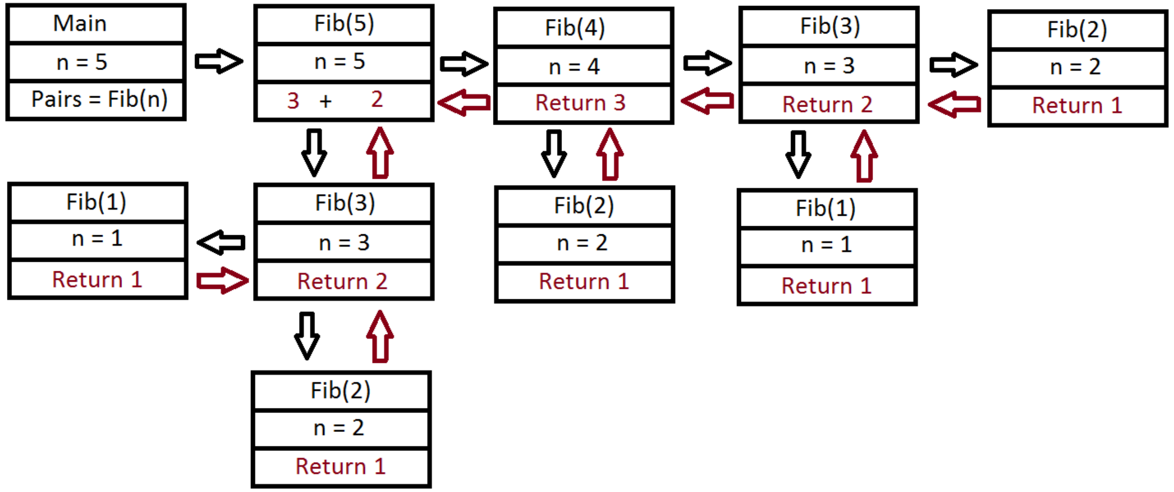
Fib(2)
n = 2
Return 1

Box Method

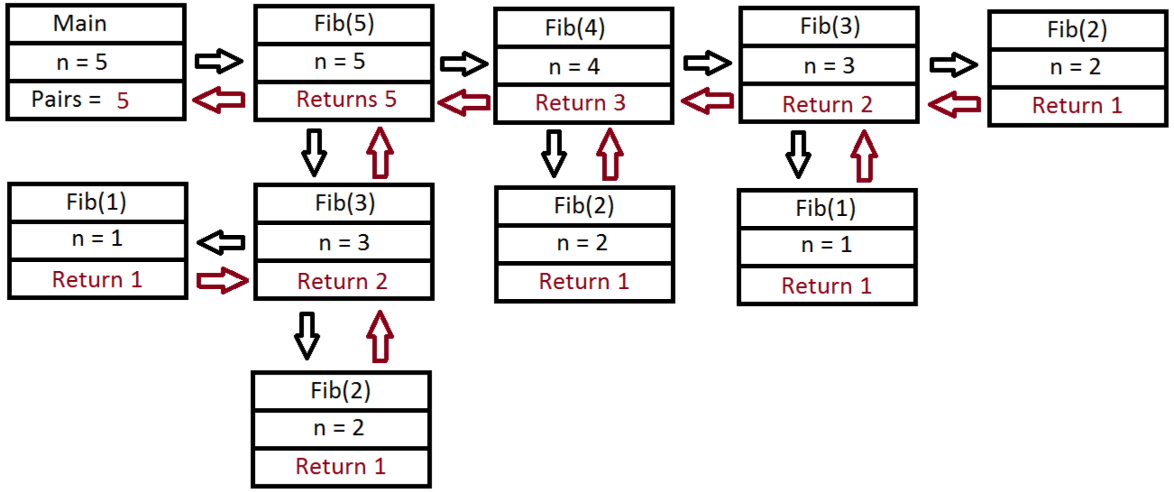


SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Box Method


Main
n = 5
Pairs = Fib(n)
Fib(5)
n = 5
3 + 2
Fib(4)
n = 4
Return 3
Fib(3)
n = 3
Return 2
Fib(2)
n = 2
Return 1
Fib(1)
n = 1
Return 1
Fib(3)
n = 3
Return 2
Fib(2)
n = 2
Return 1
Fib(1)
n = 1
Return 1
Fib(2)
n = 2
Return 1

Box Method



SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Iterative Version

```
long iterative_fib(int n)
{  int i;
   static long values[50] = {0,1,1};
   static long lastmonth = 2;
   if( n<0 )
      return 0;
   for( i=lastmonth+1; i<=n; i++)
   {
      values[i] = values[i-1] + values[i-2];
      lastmonth = n;
   }
   return values[n];
}
```

Dynamic Recursion

```
long dynamic_recursion_fib(int n)
{  static long values[52] = {0,1,1};
   static long lastmonth = 2;
   if( n<0 )
      return 0;
   if( n <= lastmonth )
      return values[n];

   values[n] = dynamic_recursion_fib(n-1) + dynamic_recursion_fib(n-2);
   lastmonth = n;
   return values[n];
}
```