Recursion

NOTES

Factorial (Code)

Iterative code (Ver. 1)

```
// Pre-cond: n >= 0
int factorial_iter1(int n)
{
    int ans = 1, i;
    for (i=1; i<=n; i++) {
        ans *= i;
    }
    return ans;
}
```

RECURSIVE

```
// Pre-cond: n >= 0
int factorial(int n)
{
    if (n == 0) // base case
        return 1;
    else
        return n * factorial(n-1);
}
```

Fibonacci (Code)

Iterative version:

```
// Pre-cond: n >= 0
int fib_iter(int n)
{
    int prev1 = 1, prev2 = 0,
        current;

    if (n < 2)
       return n;
    for (; n>1; n--) {
       current = prev1 + prev2;
       prev2 = prev1;
       prev1 = current;
    }
    return current;
}
```

Recursive version:

```
// Pre-cond: n >= 0
int fib(int n)
{
    if (n < 2) // base case
       return n;
    else
       return fib(n-1) + fib(n-2);
}
```

Sum Array (Code)

Recursive version:

Week11_SumArray.c

```
int sumArray(int arr[], int size) {
    if (size == 1)
        return arr[size-1];
    else
        return arr[size-1] + sumArray(arr, size-1);
}
```

Iterative version:

```
int sumArray_itr(int arr[], int size)
{
    int sum=0, i;

    for (i=0; i<size; i++)
        sum += arr[i];

    return sum;
}
```

Counting Occurrences

Recursive version:

```
int countValue(int value, int arr[], int size) {
   if (size == 1)
      return value == arr[0];
   else
      return (value == arr[size-1]) +
                  countValue(value, arr, size-1);
}
```

Iterative version:

```
int countValue_iter(int value, int arr[], int size)
{
    int count = 0, i;

    for (i=0; i<size; i++)
        if (value == arr[i])
            count++;

    return count;
}
```

Greatest Common Divisor

```
#include<stdio.h>
int main(){
  int n1,n2,gcd;
  printf("\nEnter two numbers: ");
  scanf("%d %d",&n1,&n2);
  gcd=findgcd(n1,n2);
  printf("\nGCD of %d and %d is: %d",n1,n2,gcd);
  return 0;
}

int findgcd(int x,int y){
     while(x!=y){
          if(x>y)
              return findgcd(x-y,y);
          else
              return findgcd(x,y-x);
     }
     return x;
}
```