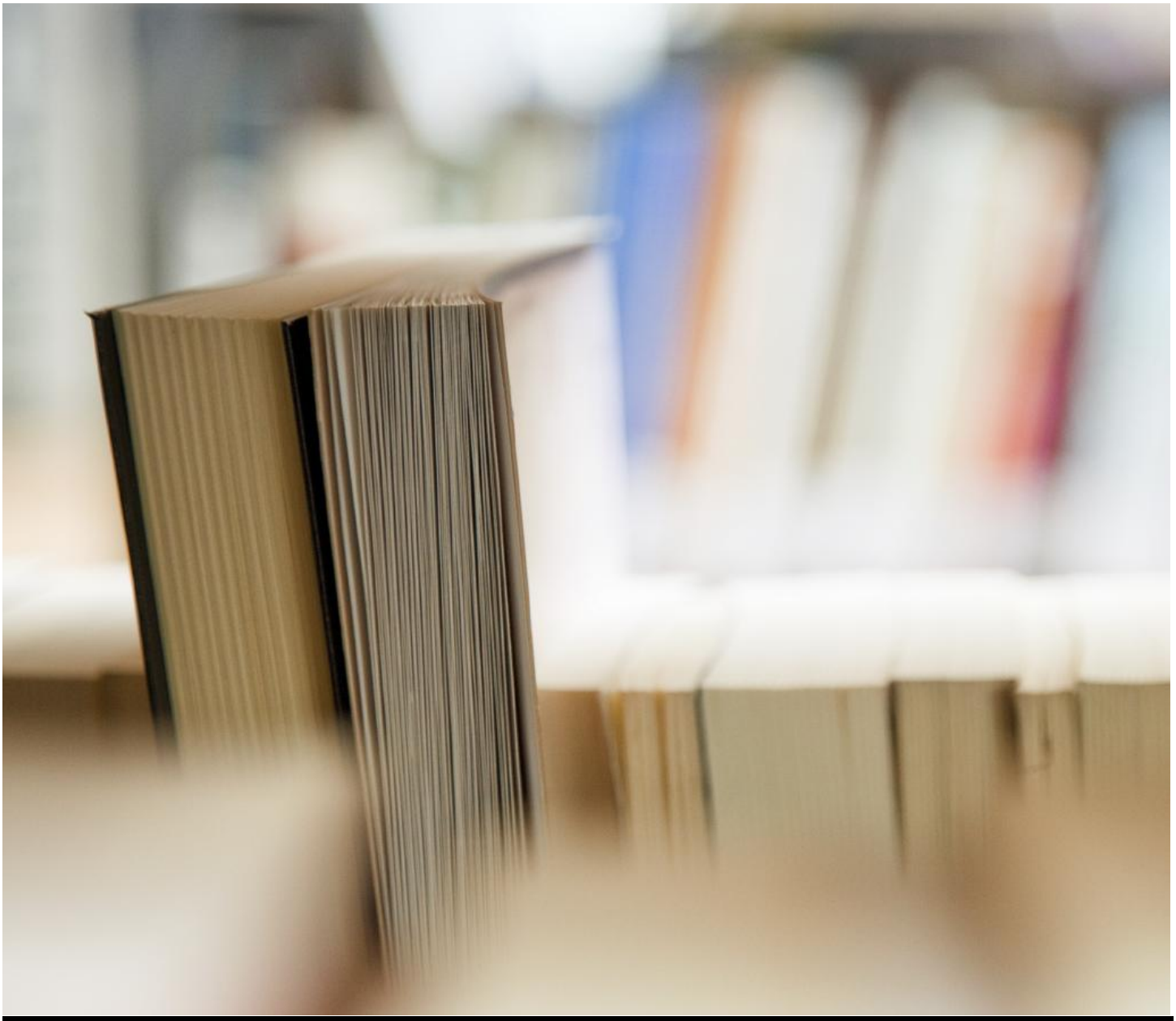

1.Recursion Examples

Data Structures and Algorithms [17ECSC204]

Prakash Hegade

2018-19

Recursion Examples

1. Find sum of N natural numbers using recursion

```
#include <stdio.h>
int sum(int num) {
    if (num!=0)
        return num + sum(num-1);
    else
        return num;
}

int main() {
    int num, result;
    printf("Enter a number: \n");
    scanf("%d", &num);
    result = sum(num);
    printf("sum=%d", result);
}
```

2. Find GCD of two numbers using recursion

```
#include <stdio.h>
int gcd(int n1, int n2)
{
    if (n2 != 0)
       return gcd(n2, n1%n2);
    else
       return n1;
}

int main()
{
   int n1, n2;
   int result;
   printf("Enter two positive integers: ");
   scanf("%d %d", &n1, &n2);

   result = gcd(n1, n2);
   printf("GCD of %d and %d is %d", n1, n2, result);
   return 0;
}
```

3. Compute factorial of a number using recursion

```
#include <stdio.h>

long long fact(int n) {
    if (n >= 1)
        return n * fact(n-1);
    else
        return 1;
}

int main() {
    int n;
    printf("Enter a number\n");
    scanf("%d", &n);
    printf("Factorial of %d = %lld", n, fact(n));
    return 0;
}
```

4. Calculate the length of a string using recursion

```
#include <stdio.h>
int getlength(char *str) {
    static int length=0;
    if(*str != '\0'){
        length++;
        str++;
        getlength(str);
    }
    else
        return length;
}

int main() {
    char str[100];
    int length=0;

    printf("Enter a string: ");
    scanf("%s", str);
    length = getlength(str);

    printf("The length is %d\n",length);
    return 0;
}
```

5. Recursive program to count the number of digits in a given number

```
#include <stdio.h>
int count_digits(int num) {
    static int count=0;
    if(num>0) {
        count++;
        count_digits(num/10);
    }
    else
        return count;
}

int main() {
    int num;
    int count=0;

    printf("Enter a number: ");
    scanf("%d",&num);
    count=count_digits(num);
    printf("Total digits in number %d is: %d\n", num, count);

    return 0;
}
```

6. Compute largest number in an array using recursion

```
#include <stdio.h>

int array_largest(int a[], int lower, int upper)
{
    int max;
    if (lower == upper)
        return a[lower];
    else {
        max = array_largest(a, lower + 1, upper);

        if (a[lower] >= max)
            return a[lower];
        else
            return max;
    }
}
```

```
int main()
{
    int a[10] = { 23, 43, 35, 38, 67, 12, 76, 10, 34, 8 };
    printf("The largest in array is %d", array_largest(a, 0, 9));
    return 0;
}
```

7. Reverse a string using recursion

```
# include <stdio.h>

void reverse(char *str) {
   if (*str) {
       reverse(str+1);
       printf("%c", *str);
   }
}

int main()
{
   char a[] = "DSA";
   reverse(a);
   return 0;
}
```

8. Nth Fibonacci number using recursion

```
#include<stdio.h>
int fib(int n)
{
   if (n <= 1)
      return n;
   return fib(n-1) + fib(n-2);
}

int main ()
{
  int n;
  printf("Enter n\n");
  scanf("%d", &n);
  printf("%d", fib(n));
  return 0;
}
```

9. Sum of array elements using recursion

```
#include <stdio.h>
int array_sum(int a[], int n)
{
    if (n <= 0)
        return 0;
    return (array_sum(a, n - 1) + a[n - 1]);
}

int main()
{
    int a[] = { 1, 2, 3, 4, 5 };
    int n = sizeof(a) / sizeof(a[0]);
    printf("%d\n", array_sum(a, n));
    return 0;
}
```

10. Check if the number is prime using recursion

```
#include<stdio.h>

int i;

int is_prime(int n)
{
    if(i==1)
        return 1;
    else if(n % i == 0)
         return 0;
    else {
         i = i - 1;
         is_prime(n);
    }
}

int main()
{

    int n, prime;

    printf("Enter a number\n");
    scanf("%d",&n);
    i = n / 2;
```

```
    prime = is_prime(n);

   if(prime == 1)
        printf("Number is Prime\n");
   else
      printf("Number is Not Prime\n");
   return 0;
}
```

~*~*~*~*~*~*~*~