RECURSION

1.Implement Power Function

```
import java.io.*;
import java.util.Scanner;

class Power {
   public static long power(int x, int n)
   {
      int pow=1;
      for (int i = 0; i < n; i++) {
         pow = pow * x;
      }

      return pow;
   }

   public static void main(String[] args)
   {
      int x;
      System.out.println("Enter value of x");
      Scanner sc=new Scanner(System.in);
      x=sc.nextInt();
      int n;
      System.out.println("Enter value of n");
      n=sc.nextInt();
      System.out.println(power(x, n));
   }
};
```

2.Sum of Digits

```
#include <stdio.h>
int sum_of_digits(int num)
{
   int temp,sum=0;
   while(num>0)
   {
      temp=num%10;
      sum=sum+temp;
      num=num/10;
   }
   return sum;

}
int main() {
   int n;//123
   printf("Enter num");
   scanf("%d",&n);
   printf("%d",sum_of_digits(n));
   return 0;
}
```

3. Is Magic

```
#include <stdio.h>
```

```
int calculateProduct(int num) {
    int product = 1;
    while (num > 0) {
        product *= num % 10;
        num /= 10;
    }
    return product;
}

int isMagic(int num, int originalNum) {
    if (num == 0) {
        return 0;
    }

    int lastDigit = num % 10;
    int product = calculateProduct(num / 10);

    if (product == lastDigit || product == originalNum) {
        return 1;
    }

    return isMagic(num / 10, originalNum);
}

int main() {
    int num;
    scanf("%d", &num);
    if (isMagic(num, num)) {
        printf("True");
    } else {
        printf("False");
    }

    return 0;
}
```

4.kth Symbol

```
#include <stdio.h>

int kth_symbol(int n, int k) {
    if (n == 1) {
        return 0;
    }

    int length = 1 << (n - 1);
    int mid = length / 2;

    if (k <= mid) {
        return kth_symbol(n - 1, k);
    } else {
        return 1 - kth_symbol(n - 1, k - mid);
    }
}
```

```
int main() {
    int n, k;
    scanf("%d %d", &n, &k);

    int result = kth_symbol(n, k);
    printf("%d",result);

    return 0;
}
```

5.Gray code

```
int* grayCode(int n, int* returnSize)
{
    *returnSize=1<<n;
      unsigned int* pAns=(unsigned int*)malloc(*returnSize*sizeof(unsigned
int));
    pAns[0]=0;
    for (unsigned int i=1;i<*returnSize;i++){
        unsigned int temp=i;
        for (int j=0;j<n;j++){
            if (temp%2!=0){
                pAns[i]=pAns[i-1]^(1<<j);
                break;
            }
            temp=temp>>1;
        }
    }
    return pAns;
}
```

6. Check Palindrome

```
#include <stdio.h>
#include <string.h>
```

```
int is_palindrome(char *inputString, int leftIndex, int rightIndex);

int main(){
   char inputString[100];
   printf("Enter a string for palindrome check\n");
   scanf("%s", inputString);

   if(isPalindrome(inputString, 0, strlen(inputString) - 1)){
      printf("True");
   } else {
      printf("False");
   }

   return 0;
}

 int isPalindrome(char *inputString, int leftIndex, int rightIndex){

    if(NULL == inputString || leftIndex < 0 || rightIndex < 0){
       printf("Invalid Input");
       return 0;
    }

    if(leftIndex >= rightIndex)
       return 1;
    if(inputString[leftIndex] == inputString[rightIndex]){
        return isPalindrome(inputString, leftIndex + 1, rightIndex - 1);
    }

    return 0;
}
```

7.Find Fibonacci-II

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

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

8. Find Factorial

```
#include<stdio.h>
```

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

int factorial(int n) {
    if (n>=1)
        return n*factorial(n-1);
    else
        return 1;
}
```

9. PrintReverse String

```
#include <stdio.h>
#include <string.h>
void reverse_string(char* str) {
    int len = strlen(str);
    if (len <= 1) {
        return;
    }
    char temp = str[0];
    str[0] = str[len-1];
    str[len-1] = temp;
    reverse_string(str+1);
}
int main() {
    char s1[] = "hello";
    reverse_string(s1);
    printf("%s\n", s1); // expected output: "olleh"
    char s2[] = "abcd";
    reverse_string(s2);
    printf("%s\n", s2); // expected output: "dcba"
    char s3[] = "";
    reverse_string(s3);
    printf("%s\n", s3); // expected output: ""
    char s4[] = "racecar";
    reverse_string(s4);
    printf("%s\n", s4); // expected output: "racecar"
    char s5[] = "12345";
    reverse_string(s5);
    printf("%s\n", s5); // expected output: "54321"
    return 0;
}
```

10.Lengthof String

```
#include <stdio.h>
int length_of_string(char *s) {
    if (*s == '\0') {
        return 0;
    } else {
        return 1 + length_of_string(s + 1);
    }
}

int main() {
    char s[] = "hello";
    printf("%d\n", length_of_string(s)); // Output: 5
    return 0;
}
```