Assignment - 2

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>

int isFeasible(char *s, int n) // checking wether the string is feasible or
{
    int count = 0;
    for(int i=0; i<(n-2); i++)
    { if(s[i]=='x' && s[i+1]=='y' && s[i+2]=='z')
        count++;
    }
    if(count==1)
      return 1;
    else
      return 0;
}
// recursively calculating all the strings
long search_exh(char *s, int n, int i, char *a)
{   if(i==n)
    return isFeasible(s,n);
    long count = 0;
    for(int k=0; k<3; k++)
    {  s[i]=a[k];
       count += search_exh(s, n, i+1, a);
    }
    return count;
}
```

```
// calculating the nCr - This will run in O(r) time
long ncr (long n, long r)
{ long res=1;
  for(int i=0; i<r; i++)
  { res = res*(n-i);
    res = res/(i+1);
  }
  return res;
}

// Calculating in polynomial time, the number of
substrings containing one "xyz"
long count_P (long n)
{ long *arr = malloc((n+1) * sizeof(long));
  long x=1;
  //precomputing the powers of 3
  // will run in O(n)
  for(int i=0; i<=n; i++)
  { arr[i]=x;
    x= x*3; // precomputing it;
  }
```

```
long *s = malloc((n+1) * sizeof(long));
for(int i=0; i<=(n-3); i++)
{  s[i] = arr[i];
   long j = (i/3);   // using it as an upper limit
   for(long k=1; k<=j; k++) // will run in O(i*i)
   { s[i] += (pow(-1,k)) * ncr(i-2*k,k) * arr[i-3*k];
   } // will run in O(n*n*n)
   long count=0;
   for(int i=0; i<=(n-3); i++)
   { count += s[i] * s[n-i-3];
   }
   return count;
}
long long search_exh_iteratively(char *s, int n, char *a)
{  long long count=0;
   long long k = pow(3,n) * 1LL;
   for(long long i=0; i<k; i++)
   { for(long long j=n-1; j>=0; j--)
     { long long x = pow(3,n-j-1) * 1LL;
       long long y = i/x;
       y = y % 3;
     } s[i] = a[y];
```

```
        count += isFeasible(s,n);
    }
    return count;
}
int main()
{   long n;
    scanf("%d", &n);
    char a[3] = "xyz";    //for comparing xyz
    char *s = malloc((n+1) * sizeof(char));
                //declaring a character array
    clock_t c1,c2;
    c1=clock();
    long count = search_exh(s,n,0,a); //calculating
        the number of substrings recursively
    c2=clock();
    printf("The Number of substrings are :");
    printf("%d\n", count);
    double exectime = (double)(c2-c1)/(double)
                                (CLOCKS_PER_SEC);
    printf("The Execution time in Seconds :");
    printf("%lf\n", exectime);
```

```
c1= clock();
long countP = count_P(n); // calculating the
    number of substrings with the formula
c2= clock();
printf(" The Number of substrings are: ");
printf("%lld\n", countP);
exectime = (double)(c2-c1)/(double)(CLOCKS_PER_S
    // clock time for polynomial
printf(" The Execution time in seconds : ");
printf("%lf \n", exectime);
char *s1= malloc((n+1) * sizeof(char));
for(int i=0; i<n; i++)
    s1[i] = 'x';
c1= clock();
long long count_iter = Search_exh_iterative(s,n,q
c2= clock();
exectime= (double)(c2-c1)/(double)(CLOCKS_PER_S
printf("The Number of substrings are :");
printf("%d\n", count_iter);
printf("The Execution time in seconds : ");
printf("%lf \n", exectime);
```