TCS NQT Model Programming/ Coding Questions Paper

1. CALCULATE LENGTH OF THE HYPOTENUSE OF RIGHT ANGLED TRIANGLE

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{

if(argc<2)
{
printf("please use \"prg_name value1 value2 ... \"\n");
return -1;
}

int a,b,side1,side2,side3;
a=atoi(argv[1]);
b=atoi(argv[2]);
side1=pow(a,2);
side2=pow(b,2);
side3=sqrt((side1+side2));
printf("the hypotenuse is %d",side3);
return 0;

}
```

2. QUES. SAMPLE PROGRAM TO PRINT ALL INTEGERS USING COMMAND LINE ARGUMENTS

```
// Program to print all value of
// command line argument
// once we get the value from command
// line we can use them to solve our problem
#include <stdio.h>

int main(int argc, char *argv[])
```

```
{
int a,b;
int i;
if(argc<2)
{
printf("please use \"prg_name value1 value2 ... \"\n");
return -1;
}

for(i=1; i<argc; i++)
{
printf("arg[%2d]: %d\n",i,atoi(argv[i]));
}


return 0;
}
```

3. FACTORIAL OF A NON NEGATIVE INTEGER

Problem Statement: Write a C program to calculate the factorial of a non-negative integer N. The factorial of a number N is defined as the product of all integers from 1 up to N. Factorial of 0 is defined to be 1. The number N is a non – negative integer that will be passed to the program as the first command line parameter. https://www.freshersnow.com/placement-papers-download/Write the output to stdout formatted as an integer WITHOUT any other additional text. You may assume that the input integer will be such that the output will not exceed the largest possible integer that can be stored in an int type variable.

FACTORIAL OF A NON-NEGATIVE INTEGER

Example:

If the argument is 4, the value of N is 4.
So, 4 factorial is 1*2*3*4 = 24.
Output : 24
The code below takes care of negative numbers but at the end of the page there is easier code which though doesn't take negative numbers in consideration. #include <stdio.h> // for printf

```
#include <stdlib.h> // for function atoi() for converting string into int
// Function to return fact value of
n int fact(int n)
{
 if (n == 0)
 return 1;
 else {
 int ans =
 1; int i;
 for (i = 1; i <= n; i++) {
 ans = ans * i;
 }
 return ans;
 }
}
// argc tells the number of arguments
// provided+1 +1 for file.exe
// char *argv[] is used to store the
// command line arguments in the string
format int main(int argc, char* argv[])
{
 // means only one argument exist that is file.exe
 if (argc == 1) {
 printf("No command line argument exist Please provide them first \n");
 return 0;
 } else {
 int i, n, ans;
 // actual arguments starts from index 1 to (argc-1)
 for (i = 1; i < argc; i++) {
 // function of stdlib.h to convert string
 // into int using atoi() function
 n = atoi(argv[i]);

 // since we got the value of n as usual of
 // input now perform operations
 // on number which you have required

 // get ans from function
 ans = fact(n);
```

```
 // print answer using stdio.h library's printf() function
 printf("%d\n", ans);
 }
 return 0;
 }
}
```

OutPut – 24 if command line argument is 4.

or simple program to write n! is –

```
#include <stdio.h>
int main(int argc, char *argv[])

{

int n,i;

unsigned long long factorial = 1;

n = atol(argv[1]);

for(i=1; i<=n; ++i)

{

factorial *= i;

}

printf(“Factorial of %d = %llu”, n, factorial);

}
```

4. WRITE A C PROGRAM TO CONVERT BINARY TO DECIMAL USING COMMAND LINE ARGUMENTS

Here is basic C program without command line –

```
#include<stdio.h>
int main(int argc, char *argv[]){
int num,binary,decimal=0,rem,base=1;
num=atoi(argv[1]);
binary=num;
while(num>0){
rem=num%2;
decimal+=rem*base;
num=num/10;
base=base*2;
}
printf("%d",decimal);
return 0;
}
```

5. WRITE A C PROGRAM TO CHECK WHETHER GIVEN NO. IS PALINDROME OR NOT USING COMMAND LINE ARGUMENTS

```
#include<stdio.h>
#include<stdlib.h>
int main(int argc, char* argv[])
{
 int num=atoi(argv[1]);
 if(isPalindrome(num))
 printf("Palindrome");
 else
 printf("Not Palindrome");

 return 0;
}
int isPalindrome(int n)
{
 int m=n;
 int rev=0;
 while(m!=0)
 {
 rev=(rev*10) + (m%10);
 m=m/10;
 }
```

```
 if(rev==n)
 return 1;
 else
 return 0;
}
```

6. Write a C program that will find the sum of all prime numbers in a given range.The range will be specified as command line parameters. The first command line parameter, N1 which is a positive integer, will contain the lower bound of the range. The second command line parameter N2, which is also a positive integer will the upper bound of the range. The program should consider all the prime numbers within the range, excluding the upper and lower bound. Print the output in integer format to stdout. Other than the integer number, no other extra information should be printed to stdout.

```
#include<stdio.h>
int main(int argc,char *argv[])
{
int N1,N2,i,j,sum=0,count,lower,upper;
if(argc!=3)
 exit(0);
N1=atoi(argv[1]);
lower=N1+1;
N2=atoi(argv[2]);
upper=N2;
 for(i=lower;i<upper;i++)
 {
 count=1;
 for(j=2;j<=i/2;j++)
 {
 if(i%j==0)
 {
 count++;
 }
 }
 if(count==1)
 {
 sum=sum+i;
 }
 }
 printf("%d",sum);
```

```
return 0;
}
```