Assignment - 12 (Recursion in C Language)

1. Write a recursive function to print first N natural numbers

void printFirstNNaturalNums(int);

void printFirstNNaturalNums(int n)

{

if(n == 1)

printf("1 ");

else

{

printFirstNNaturalNums(n - 1);

printf("%d ", n);

}

}

2. Write a recursive function to print first N natural numbers in reverse order

void printFirstNNaturalNums\_ReverseOrder(int);

void printFirstNNaturalNums\_ReverseOrder(int n)

{

if(n == 1)

printf("1 ");

else

{

printf("%d ", n);

printFirstNNaturalNums\_ReverseOrder(n - 1);

}

}

3. Write a recursive function to print first N odd natural numbers

void firstNOddNaturalNums(int);

void firstNOddNaturalNums(int n)

{

if(n == 1)

printf("1 ");

else

{

firstNOddNaturalNums(n - 1);

printf("%d ", 2 \* n - 1);

}

}

4. Write a recursive function to print first N odd natural numbers in reverse order

void firstNOddNaturalNums\_inReverse(int);

void firstNOddNaturalNums\_inReverse(int n)

{

if(n == 1)

printf("1 ");

else

{

printf("%d ", 2 \* n - 1);

firstNOddNaturalNums\_inReverse(n - 1);

}

}

5. Write a recursive function to print first N even natural numbers

void printFirstNEvenNaturalNums(int);

void printFirstNEvenNaturalNums(int n)

{

if(n == 1)

printf("2 ");

else

{

printFirstNEvenNaturalNums(n - 1);

printf("%d ", n \* 2);

}

}

6. Write a recursive function to print first N even natural numbers in reverse order

void printFirstNEvenNaturalNums\_inReverse(int);

void printFirstNEvenNaturalNums\_inReverse(int n)

{

if(n == 1)

printf("2 ");

else

{

printf("%d ", n \* 2);

printFirstNEvenNaturalNums\_inReverse(n - 1);

}

}

7. Write a recursive function to print squares of first N natural numbers

void printSquaresOfFirstNNaturalNums(int);

void printSquaresOfFirstNNaturalNums(int n)

{

if(n == 1)

printf("1 ");

else

{

printSquaresOfFirstNNaturalNums(n - 1);

printf("%d ", n \* n);

}

}

8. Write a recursive function to print binary of a given decimal number

void floating\_binaryOfDecimal(float);

void floating\_binaryOfDecimal(float n)

{

int i;

for(i = 1; i <= 8; i++)

{

n = n \* 2;

if(n >= 1)

{

int integral = n;

n = n - integral;

printf("%d", integral);

if(n == 0)

break;

}

else

printf("0");

}

}

void print\_binaryOfDecimal(float);

void print\_binaryOfDecimal(float n)

{

if(n == 0)

printf("0");

else if(n < 0)

{

printf("-");

print\_binaryOfDecimal(n \* -1);

}

else

{

int integral = n;

if(integral == n)

{

print\_binaryOfDecimal(integral / 2);

printf("%d", integral % 2);

}

else

{

float fraction = n - integral;

print\_binaryOfDecimal(integral);

printf(".");

floating\_binaryOfDecimal(fraction);

}

}

}

9. Write a recursive function to print octal of a given decimal number

void floating\_octalOfDecimal(float);

void floating\_octalOfDecimal(float n)

{

int i;

for(i = 1; i <= 8; i++)

{

n = n \* 8;

if(n >= 1)

{

int integral = n;

n = n - integral;

printf("%d", integral);

if(n == 0)

break;

}

else

printf("0");

}

}

void print\_octalOfDecimal(float);

void print\_octalOfDecimal(float n)

{

if(n == 0)

printf("0");

else if(n < 0)

{

printf("-");

print\_octalOfDecimal(n \* -1);

}

else

{

int integral = n;

if(integral == n)

{

print\_octalOfDecimal(integral / 8);

printf("%d", integral % 8);

}

else

{

float fraction = n - integral;

print\_octalOfDecimal(integral);

printf(".");

floating\_octalOfDecimal(fraction);

}

}

}

10. Write a recursive function to print reverse of a given number

void reverse\_num(int);

void reverse\_num(int n)

{

if(n == 0)

printf("");

else

{

printf("%d", n % 10);

reverse\_num(n / 10);

}

}