Lab6: Recursion + Structures

Instructions

- Make your own file named Submission.cpp. Please don't include the main function while submitting the file.
- Data member names of each structure should be the same as per mentioned in each question.
- Please read the questions carefully, read them twice even thrice to understand them completely.
- Please see the corresponding function signature in the skeleton file to further understand the question.
- In case of any query, please raise your hands and we will be there to solve your query.
- Please concentrate, understand and code. Good Luck :)

Task 1

Write a function **magicSum** that calculates and returns ((1 + 2 … + x-1 + x) +y) which is x(x+1)/2 + y. For example if x is 5 and y is 2, then function should return 15 + 2 = 17.

Task 2

Write a recursive function Collatz to print Collatz sequence for a given positive integer. The Collatz sequence states that every positive integer eventually reaches unity if we apply following partial function iteratively to the number

For any positive integer n

F (n) = $\begin{cases} \frac{n}{2} & if\ n\ is\ even \\ 3n+1 & if\ n\ is\ odd \end{cases}$

Example: A call to Collatz(7) prints the sequence

7 22 11 34 17 52 26 13 40 20 10 5 16 8 4 2 1

Task 3

Spherical objects can be stacked to form a pyramid with one cannonball at the top, sitting on top of a square composed of four cannonballs, sitting on top of a square composed of nine cannonballs, and so forth. Write a recursive method **countCannonballs** that takes as its

argument the height of a pyramid of cannonballs and returns the total number of cannonballs that the stack contains.
For example, a call to countCannonballs(3) returns 14

Task 4

Write a recursive function **calculateGcd** which take two integer arguments and calculate their Greatest Common Divisor (GCD).
Example: a call to calculateGcd (12, 90) returns 6

Task 5

Write a recursive function **parseToInteger** which takes a string input (which represents an integer) as an argument and returns its integer value.
For example a call to parseToInteger("32") should return 32
Hint: The ASCII value of char '0' is 48

Task 6

Write a recursive function **PrintPattern1** that print the following pattern on screen. The function will take a single argument of type integer as input.
Hint: You may use another recursive function/loop to help print the contents of each row.

```
* * * * *
* * * *
* * *
* *
*
*
* *
* * *
* * * *
* * * * *
```

Task 7

Write a recursive function **PrintPattern2** that print the following pattern on screen. The function will take a single argument of type integer as input.

```
*******
 *****
  ***
   *
   *
  ***
 *****
*******
```

TASK 8

a) A phone number can be thought of three parts: the area code (111), the exchange (767), and the number (1011). All data member are of type int. Write a structure named as PhoneNumber to store these three parts of phone number. Data members should be named as areaCode of type int,exchange of type int and number of type int .

b) Write a function PhoneNumberArrayIntialize which takes an array arr of type PhoneNumber, of size s. You have to store PhoneNumber objects at each index of array arr. Assign values to each object of PhoneNumber such that for first object area code starts from 111, exchange starts from 767 and number starts from 1011 while for the second object area code is 112, exchange is 768 and number 1012, and it continues for all objects sequentially of size s.

   Void PhoneNumberArrayIntialize(PhoneNumber arr[],int s)

c) Write a function PrintPhoneNumber which takes p as PhoneNumber object ,you have to print p in following format
   (area code) Exchange-Number .
   Example: (111) 767-1011

   Void PrintPhoneNumber(PhoneNumber p)

TASK 9

Define a structure to represent a bank account named as **BankAccount** that includes the following members:

Private Data Members:

- depositorName of type string.
- accountNumber of type string
- accountCat of type **accountCategory** (having data members, int Id, string Name)
- Balance of type long

Public member functions:

- To initialize the **BankAccount** structure with its required values, you need to use the following function
  void Initialize (string dp, string an, int acountTypeId, string accountTypeName, long balance)
- To deposit an amount you need to use the following function
  bool depositAmount(long amountToDeposite)
  Hint: you have to validate that amount to deposit is non-negative.
- To withdraw an amount after checking the balance. Return False if amount is less than with drawl amount otherwise return True in case of successful transaction.
  bool withdrawAmount(long amountToWithdraw)
- To get balance in given accountNum, long getAmount()