**Name: Veeransh Shah**

**Reg No.: 221070063**

**Experiment No. 2**

**To add two large integers using stacks**

**AIM: To add 2 large integers using stacks by operator overloading**

**THEORY:**

Long Integer Arithmetic Addition involves adding two integers that may exceed the capacity of standard integer data types. To perform this operation, a stack-based approach can be employed. The theory below outlines the key concepts and steps involved in implementing long integer addition using a stack.

1. Data Representation:

- The long integers are represented as strings, where each digit is a separate element in the string. For example, the number 1234 is represented as "1234" in the string.

2. Stack Implementation:

- A stack data structure is used to facilitate the addition operation. The stack allows us to process the digits from right to left, mimicking the manual addition process.

**Algorithm:**

1. Start
2. Create Class Stack
3. Check if the stack is underflowing or overflowing
4. Create functions of push and pull
5. Create function Peek – The value at the top is displayed
6. Create function Display – to display elements in the stack
7. Create Main Function and give all the commands

**Example:-**

lets take 2 numbers

99+13

->99 is put on stack 1 in order as 9,9

and 13 is put on stack 2 in order as 1,3

-> 9 from stack1 and 3 from stack 2 is popped and added it gives 12.

12%10 i.e. 2 is pushed onto stack 3

and carry equals 12/10 i.e 1

->now 9 and 1 is popped off stack 1 and stack 2 and added to carry which is 1

sum= 9 +1 +1 =11

11%10 =1 is pushed onto stack 3 and carry =11/10=1 which is not 0 and both stacks 1 and 2 are empty, hence it is pushed onto stack 3.

->now ans is pushed in stack in the order 2,1,1.

->now while printing on the screen we pop elements of stack 3 while it is not empty and print it on the screen. hence on the screen it is displayed as 112 .

**CONCLUSION:**

Thus, from this experiment we have gained knowledge of stacks and successful added two large number using stacks.

**ScreenShot:**

