DATE:

Name: Prerna Sunil Jadhav
SapID: 60004220127
Batch: C22
Course: Advance Algorithm Lab

EXP 1B

AIM: Perform Amortized Analysis using Accounting method.

THEORY: Amortized Analysis is a method used to analyze the performance of algorithms that perform a sequence of operations, where each individual operation may be fast, but the sequence of operation may be slow as a whole. It is used to determine the average cost per operation, allowing for a more accurate comparison of algorithm that perform different number of operations.

ACCOUNTING METHOD:

→ It can useful in understanding the performance of algorithm that performs a sequence of operation with varying cost.

→ It can be applied to a wide range of data structure and algorithms.

→ Unlike the aggregated analysis, the accounting method assigns a different cost to each type of operation.

DATE:

→ The accounting method is much like managing your personal finances; you can estimate the cost of your operations however you like, as long as, at the end of the day, the amount of money you have set aside is enough to pay bills.

→ The estimate cost of an operation may be greater or less than its actual cost; correspondingly the surplus of one operation can be used to pay the debt of other operations.

CONCLUSION: Thus we studied about the accounting method in Amortized analysis.

Name:	Prerna Sunil Jadhav
Sap Id:	60004220127
Class:	T. Y. B. Tech (Computer Engineering)
Course:	Advance Algorithm Laboratory
Course Code:	DJ19CEL602
Experiment No.:	01-B

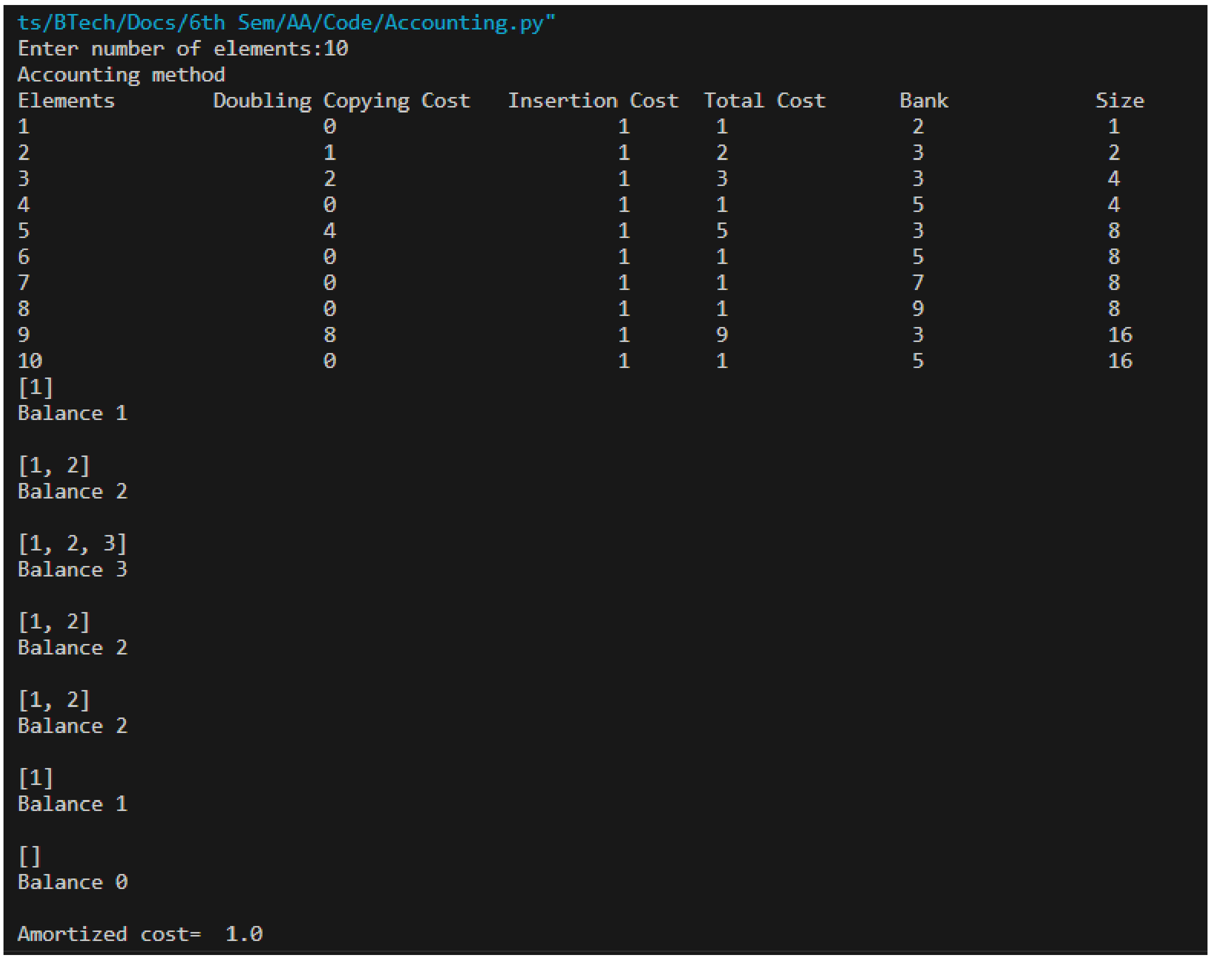
AIM: **Perform Amortized Analysis of Multipop / Dynamic Tables / Binary Counter using Aggregate, Accounting and Potential method. (Amortized Analysis)**

1B) Amortized Analysis (Accounting method)

CODE:

```python
def accounting(n):
    size=1
    total=0
    dcost=0
    icost=0
    bank=0

    print("Elements\tDoubling Copying Cost\tInsertion Cost\tTotal
Cost\tBank\t\tSize")

    for i in range(1,n+1):

        icost=1
        if i>size:
            size*=2
            dcost=i-1

        total=icost+dcost
        bank+=(3-total)

        print(i,"\t\t\t",dcost,"\t\t\t",icost,"\t",total,"\t\t",bank,"\t\t",si
ze)
        icost=0
        dcost=0

n=int(input("Enter number of elements:"))
print("Accounting method")
accounting(n)

class AccountingStack:

    def __init__(self):
        self.stack=[]
        self.cost=0
        self.balance=0
    def push(self,item):
        self.stack.append(item)
        self.cost+=1
        self.balance+=1
        self.printstack()

    def pop(self):
        self.stack.pop()
        self.cost+=1
        self.balance-=1
        self.printstack()

    def multipop(self,k):
        for i in range(k):
            self.pop()

    def printstack(self):
        print(self.stack,"\nBalance",self.balance,"\n")

s=AccountingStack()

s.push(1)
s.push(2)
s.push(3)

s.pop()

s.printstack()
s.multipop(2)

print("Amortized cost= ",s.cost/6)
```

OUTPUT:



```
ts/BTech/Docs/6th Sem/AA/Code/Accounting.py"
Enter number of elements:10
Accounting method
Elements        Doubling Copying Cost   Insertion Cost  Total Cost      Bank            Size
1                        0                        1       1              2               1
2                        1                        1       2              3               2
3                        2                        1       3              3               4
4                        0                        1       1              5               4
5                        4                        1       5              3               8
6                        0                        1       1              5               8
7                        0                        1       1              7               8
8                        0                        1       1              9               8
9                        8                        1       9              3               16
10                       0                        1       1              5               16
[1]
Balance 1

[1, 2]
Balance 2

[1, 2, 3]
Balance 3

[1, 2]
Balance 2

[1, 2]
Balance 2

[1]
Balance 1

[]
Balance 0

Amortized cost=  1.0
```

CONCLUSION: Hence we studied amortized analysis-Accounting method.