Name	Ronak Thakar
UID no.	2021700066
Experiment No.	6

AIM:	Greedy Method – Prim's Algorithm.
Program	
PROBLEM STATEMENT:	Use Greedy Programming method to find the minimum weight of a spanning tree.
ALGORITHM/ THEORY:	Greedy is an algorithmic paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit. Prim's algorithm is an example of greedy algorithm. This algorithm always starts with a single node and moves through several adjacent nodes, in order to explore all the connected edges along the way. The algorithm starts with an empty spanning tree. The idea is to maintain two sets of vertices. The first set contains the vertices already included in the MST, and the other set contains the vertices not yet included. At every step, it considers all the edges that connect the two sets and picks the minimum weight edge from these edges. After picking the edge, it moves the other endpoint of the edge to the set containing MST. Steps to implement Prim's algorithm: Step 1: Determine an arbitrary vertex as the starting vertex of the MST. Step 2: As mentioned above we keep trace of the vertices as visited and not-visited i.e. the vertices are included in the MST or not. Taking this into account we follow the steps 3 to 5 till all the vertices are in the MST. Step 3: Find the edges that connects the current MST with the fringe vertices. Step 4: Find the edge that costs us the minimum weight. Step 5: Add the chosen edge to the MST if it does not form a cycle. Step 6: Print the MST and Stop. Advantages of Prim's algorithm: • It is guaranteed to find the MST in a connected, weighted graph. • It has a time complexity of $O(V^2)$ when implemented using arrays

and a time complexity of O(E*logV) when implemented using a binary heap or Fibonacci heap.

- It is relatively simple and easy to understand and implement the algorithm.

Disadvantages of Prim's algorithm:

- It can be slow on dense graphs with many edges, as it requires iterating over all the edges at least once.
- Prim's algorithm relies on a priority queue, which can take up extra memory and slow down the algorithm on very large graphs.
- The choice of starting node can affect the MST output.

PROGRAM:

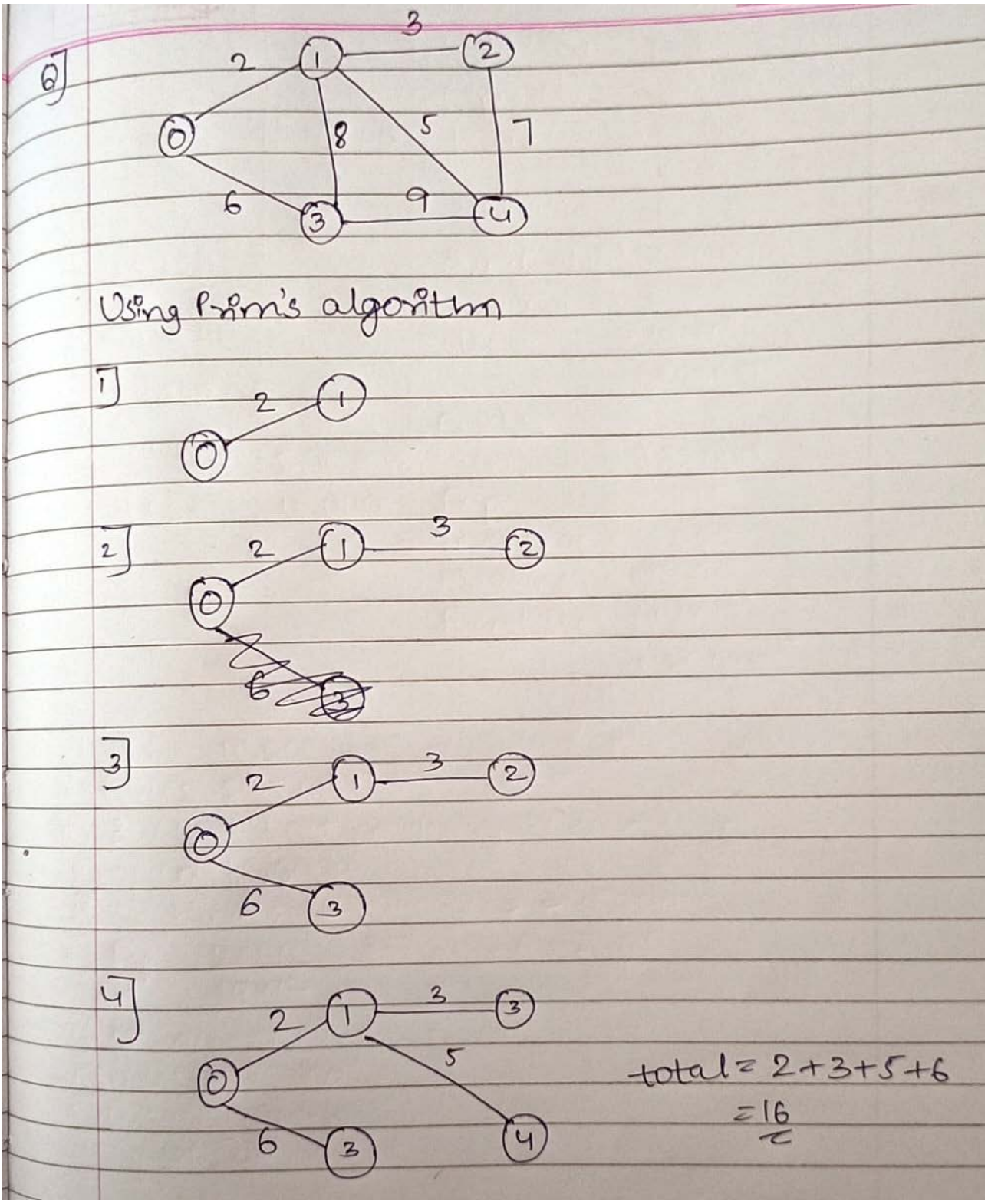
```
#include<stdio.h>
#include<stdbool.h>
int parent[100],weight[100][100];
int min(int key[],bool isused[],int num){
   int minkey = 20000,i;
   for(int x=0;x<num;x++){
      if(isused[x] == false && key[x] < minkey){
         minkey = key[x];
         i = x;
      }
   }
   return i;
}
int prim(int num){
   int key[num],total=0;
   bool isused[num];
   for(int i=0;i<num;i++){
      key[i] = 20000;
      isused[i] = false;
   }
   key[0] = 0;
   parent[0] = -1;
   for(int i=0;i<num-1;i++){
      int u = min(key,isused,num);
      isused[i] = true;
      for(int x=0;x<num;x++){
         if(weight[u][x] < key[x] && isused[x] == false &&
weight[u][x]>0){
            parent[x] = u;
            key[x] = weight[u][x];
            total = total + key[x];
         }
      }
   }
```

```
    return total;
}
int main(){
    int num,s,e,w,ch;
    printf("\nEnter the number of vertices: ");
    scanf("%d",&num);
    printf("\nEnter the edges and weights: ");
    do{
        printf("\nEnter Start Point: ");
        scanf("%d",&s);
        printf("\nEnter End Point: ");
        scanf("%d",&e);
        printf("\nEnter Weight: ");
        scanf("%d",&w);
        weight[s][e] = w;
        weight[e][s] = w;
        printf("\nEnter 0 to stop: ");
        scanf("%d",&ch);
    }while(ch!=0);
    int total = prim(num);
    printf("Edge\tWeight");
    for(int i=1;i<num;i++){
      printf("\n%d - %d\t%d",parent[i],i,weight[i][parent[i]]);
    }
    printf("\nTotal = %d",total);
}
```

RESULT:

```
Enter the number of vertices: 5

Enter the edges and weights:
Enter Start Point: 0

Enter End Point: 1

Enter Weight: 2

Enter 0 to stop: 1

Enter Start Point: 0

Enter End Point: 3

Enter Weight: 6

Enter 0 to stop: 1

Enter Start Point: 1

Enter End Point: 2

Enter Weight: 3

Enter 0 to stop: 1
```

```
Enter Start Point: 1

Enter End Point: 3

Enter Weight: 8

Enter 0 to stop: 1

Enter Start Point: 1

Enter End Point: 4

Enter Weight: 5

Enter 0 to stop: 1

Enter Start Point: 2

Enter End Point: 4

Enter Weight: 7

Enter 0 to stop: 1

Enter Start Point: 3

Enter End Point: 4

Enter Weight: 9

Enter 0 to stop: 0
Edge    Weight
0 - 1   2
1 - 2   3
0 - 3   6
1 - 4   5
Total = 16
```



CONCLUSION:	We studied what is greedy approach and how to implement it using the prim's algorithm.