ADVANCED ALGORITHMS

Final Assessment Test

LAB EXAM

NAME: D.NANDHA KUMAR

REG.NO: 19MIC0019

SLOT: L15 + L16

QUESTION-1:

CODE:

```
#include <iostream>
#include <numeric>
using namespace std;

bool checkSum(int sumLeft[], int k)
{
    int r = true;
    for (int i = 0; i < k; i++)
    {
        if (sumLeft[i] != 0) {
            r = false;
        }
    }

    return r;
}
```

```
bool subsetSum(int S[], int n, int sumLeft[], int A[], int k)
{
    if (checkSum(sumLeft, k)) {
        return true;
    }
     if (n < 0) {
        return false;
    }
    bool result = false;
    for (int i = 0; i < k; i++)
    {
        if (!result && (sumLeft[i] - S[n]) >= 0)
        {

            A[n] = i + 1;
            sumLeft[i] = sumLeft[i] - S[n];
            result = subsetSum(S, n - 1, sumLeft, A, k);
            sumLeft[i] = sumLeft[i] + S[n];
        }
    }
    return result;
}

void partition(int S[], int n, int k)
{
```

```
if (n < k)
{
    cout << "k-partition of set S is not possible";
    return;
}


int sum = accumulate(S, S + n, 0);

int A[n], sumLeft[k];


for (int i = 0; i < k; i++) {
    sumLeft[i] = sum/k;
}

bool result = !(sum % k) && subsetSum(S, n - 1, sumLeft, A, k);

if (!result)
{
    cout << "k-partition of set S is not possible";
    return;
}

for (int i = 0; i < k; i++)
{
```

```
        cout << "Partition " << i << " is ";
        for (int j = 0; j < n; j++)
        {
            if (A[j] == i + 1) {
                cout << S[j] << " ";
            }
        }
        cout << endl;
    }
}

int main()
{
    int n;
    cout<<"\nNAME:D.Nandha kumar\nREG.NO:19MIC0019\nEnter the number
of elements in the multiset(must be divisble by 3): ";
    cin>>n;
    int S[n];
    cout<<"\nEnter the elemets in the multiset\n\n";
    for(int i=0;i<n;i++)
        cin>>S[i];
    int k = n/3;
    cout<<"\n";
    partition(S, n, k);
    cout<<"\n";
    return 0;
}
```

OUTPUT:

```
C:\Users\NANDHAKUMAR\Documents\adv fat-1.exe

NAME:D.Nandha kumar
REG.NO:19MIC0019
Enter the number of elements in the multiset(must be divisble by 3): 12

Enter the elemets in the multiset

20 23 25 30 45 45 27 30 30 40 22 23

Partition 0 is 45 22 23
Partition 1 is 23 27 40
Partition 2 is 30 30 30
Partition 3 is 20 25 45


--------------------------------
Process exited after 108.4 seconds with return value 0
Press any key to continue . . .
```

QUESTION-2:

CODE:

```
#include<stdio.h>

#include<stdlib.h>

int arr[100][100];

int cad[100];

int blue[100];

int red[100];

int main()

{

        printf("\nNAME:D.Nandha kumar\n");

   printf("\nREG.NO:19MIC0019\n");
```

```
int n;
int m;
int b;
int r;
printf("enter the number of vertices:");
scanf("%d",&n);
printf("enter the adjacency matrix:");
for(int i=0;i<n;i++)
{
   for(int j=0;j<n;j++)
   {
      scanf("%d",&arr[i][j]);
   }
}
printf("enter the number of elements in V':");
scanf("%d",&m);
printf("enter the elements of V':");
for(int i=0;i<m;i++)
{
   scanf("%d",&cad[i]);
}
printf("enter the number of elements in set of blue vertices:");
scanf("%d",&b);
printf("enter the elements in blue set:");
```

```
for(int i=0;i<b;i++)
{
   scanf("%d",&blue[i]);
}
printf("enter the number of elements in set of red vertices:");
scanf("%d",&r);
printf("enter the elements in red set:");
for(int i=0;i<r;i++)
{
   scanf("%d",&red[i]);
}
for(int i=0;i<m;i++)
{
   for(int j=0;j<n;j++)
   {
      if(arr[cad[i]-1][cad[i+1]-1]==1)
      {
         printf("it should not have adjacent edges");
         exit(0);
      }
   }
}
for(int i=0;i<n;i++)
{
```

```
        for(int j=0;j<n;j++)
        {
            if(cad[i]==blue[i])
            {
                for(int k=0;k<r;k++)
                {
                    if(arr[cad[i]-1][k]==1)
                    {
                        for(int k1=0;k1<r;k1++)
                        {
                            if(arr[cad[i]-1][k]==blue[k1])
                            {
                                printf("Connection should not be between same
sets:");
                                exit(0);
                            }
                        }
                    }
                }
            }
            if(cad[i]==red[i])
            {
                for(int k=0;k<r;k++)
                {
```

```
                if(arr[cad[i]-1][k]==1)
                {
                    for(int k1=0;k1<r;k1++)
                    {
                        if(arr[cad[i]-1][k]==red[k1])
                        {
                            printf("Connection should not be between same
sets:");
                            exit(0);
                        }
                    }
                }
            }
        }
    }
    printf("THE GIVEN SUBSET IS A APRROXIMATION SOLUTION");
    for(int i=0;i<m;i++)
    {
        printf("%d\t",cad[i]);
    }
}
```

OUTPUT:

C:\Users\NANDHAKUMAR\Documents\adv fat-2.exe

```
NAME:D.Nandha kumar

REG.NO:19MIC0019
enter the number of vertices:8
enter the adjacency matrix:
0 1 1 0 1 0 0 0
1 0 0 1 0 1 0 0
1 0 0 1 0 0 1 0
0 1 1 0 1 0 0 0
1 0 0 0 0 1 1 0
0 1 0 0 1 0 0 1
0 0 1 0 1 0 0 1
0 0 0 1 0 1 1 0
enter the number of elements in V':4
enter the elements of V':2 3 5 6
enter the number of elements in set of blue vertices:4
enter the elements in blue set:2 3 5 6
enter the number of elements in set of red vertices:4
enter the elements in red set:1 4 7 8
it should not have adjacent edges
--------------------------------
Process exited after 148.8 seconds with return value 0
Press any key to continue . . .
```