Name: Devansh Upadhyay

Roll No: 53

Program:

```c
#include <stdio.h>
#include <stdlib.h>
#define INFINITY 99999
struct Edge {
  int u;
  int v;
  int w;
};
struct Graph {
  int V;
  int E;
  struct Edge *edge;
};
void bellmanford(struct Graph *g, int source);
void display(int arr[], int size);
int main(void) {
  struct Graph *g = (struct Graph *)malloc(sizeof(struct Graph));
  g->V = 4;
  g->E = 5;
  g->edge = (struct Edge *)malloc(g->E * sizeof(struct Edge));
  g->edge[0].u = 0;
  g->edge[0].v = 1;
  g->edge[0].w = 5;
  g->edge[1].u = 0;
  g->edge[1].v = 2;
  g->edge[1].w = 4;
  g->edge[2].u = 1;
  g->edge[2].v = 3;
  g->edge[2].w = 3;
  g->edge[3].u = 2;
  g->edge[3].v = 1;
  g->edge[3].w = 6;
  g->edge[4].u = 3;
  g->edge[4].v = 2;
  g->edge[4].w = 2;
  bellmanford(g, 0);
  return 0;
}
```

```c
void bellmanford(struct Graph *g, int source) {
  int i, j, u, v, w;
  int tV = g->V;
  int tE = g->E;
  int d[tV];
  int p[tV];
  for (i = 0; i < tV; i++) {
    d[i] = INFINITY;
    p[i] = 0;
  }d[source] = 0;
  for (i = 1; i <= tV - 1; i++) {
    for (j = 0; j < tE; j++) {
      u = g->edge[j].u;
      v = g->edge[j].v;
      w = g->edge[j].w;
      if (d[u] != INFINITY && d[v] > d[u] + w) {
        d[v] = d[u] + w;
        p[v] = u;
      }}}
  for (i = 0; i < tE; i++) {
    u = g->edge[i].u;
    v = g->edge[i].v;
    w = g->edge[i].w;
    if (d[u] != INFINITY && d[v] > d[u] + w) {
      printf("Negative weight cycle detected!\n");
      return;
    }}
  printf("Distance array: ");
  display(d, tV);
  printf("Predecessor array: ");
  display(p, tV);
}
void display(int arr[], int size) {
  int i;
  for (i = 0; i < size; i++) {
    printf("%d ", arr[i]);
  }
  printf("\n");
}
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

PS D:\5th-Sem-Practicals\DAA\Pract 8> gcc prog.c
PS D:\5th-Sem-Practicals\DAA\Pract 8> ./a.exe
Distance array: 0 5 4 8
Predecessor array: 0 0 0 1
PS D:\5th-Sem-Practicals\DAA\Pract 8>
```