**Name: Saqib Shehzad**

**Registration No. : 2021-CS-187**

**Submitted to Mam Abqa Javed**

**OS Lab Mid Paper.**

Task:

Code:

#include <stdio.h>

// Maximum number of accounts and resources

#define MAX\_ACCOUNTS 10

#define MAX\_RESOURCES 5

// Available resources (G, Fixed deposit, Monthly Income Scheme, Gold, etc.)

int available[MAX\_RESOURCES];

// Maximum resources that can be allocated to each account

int maximum[MAX\_ACCOUNTS][MAX\_RESOURCES];

// Resources currently allocated to each account

int allocation[MAX\_ACCOUNTS][MAX\_RESOURCES];

// Remaining resources needed by each account

int need[MAX\_ACCOUNTS][MAX\_RESOURCES];

// Array to track finished accounts

int finished[MAX\_ACCOUNTS];

// Number of accounts and resources

int num\_accounts, num\_resources;

// Function to check if an account can be processed

int is\_safe(int account)

{

int i, j;

int work[MAX\_RESOURCES];

// Initialize work array

for (i = 0; i < num\_resources; i++) {

work[i] = available[i];

}

// Check if account can be finished

for (i = 0; i < num\_resources; i++) {

if (need[account][i] > work[i]) {

return 0;

}

}

// Simulate allocation of resources to account

for (i = 0; i < num\_resources; i++) {

work[i] += allocation[account][i];

}

// Check if all accounts can be finished

for (i = 0; i < num\_accounts; i++) {

if (!finished[i]) {

for (j = 0; j < num\_resources; j++) {

if (need[i][j] > work[j]) {

break;

}

}

if (j == num\_resources) {

finished[i] = 1;

for (j = 0; j < num\_resources; j++) {

work[j] += allocation[i][j];

}

}

}

}

// Check if account can be finished

if (finished[account]) {

return 1;

} else {

return 0;

}

}

int main()

{

int i, j;

// Input number of accounts and resources

printf("Enter number of accounts: ");

scanf("%d", &num\_accounts);

printf("Enter number of resources: ");

scanf("%d", &num\_resources);

// Input available resources

printf("Enter available resources:\n");

for (i = 0; i < num\_resources; i++) {

scanf("%d", &available[i]);

}

// Input maximum resources for each account

printf("Enter maximum resources for each account:\n");

for (i = 0; i < num\_accounts; i++) {

printf("Account %d:\n", i);

for (j = 0; j < num\_resources; j++) {

scanf("%d", &maximum[i][j]);

}

}

// Input resources currently allocated to each account

printf("Enter resources currently allocated to each account:\n");

for (i = 0; i < num\_accounts; i++) {

printf("Account %d:\n", i);

for (j = 0; j < num\_resources; j++) {

scanf("%d", &allocation[i][j]);

need[i][j] = maximum[i][j] - allocation[i][j];

}

}

// Calculate remaining resources needed by each account

for (i = 0; i < num\_accounts; i++) {

for (j = 0; j < num\_resources; j++) {

need[i][j] = maximum[i][j] - allocation[i][j];

}

}

// Check if car loan can be processed

for (i = 0; i < num\_accounts; i++) {

if (is\_safe(i)) {

printf("Account %d can be processed.\n", i);

return 0;

}

}

printf("Car loan cannot be processed.\n");

return 0;

}

**Output:**

