EX.N0:03
DATE:29.01.2024

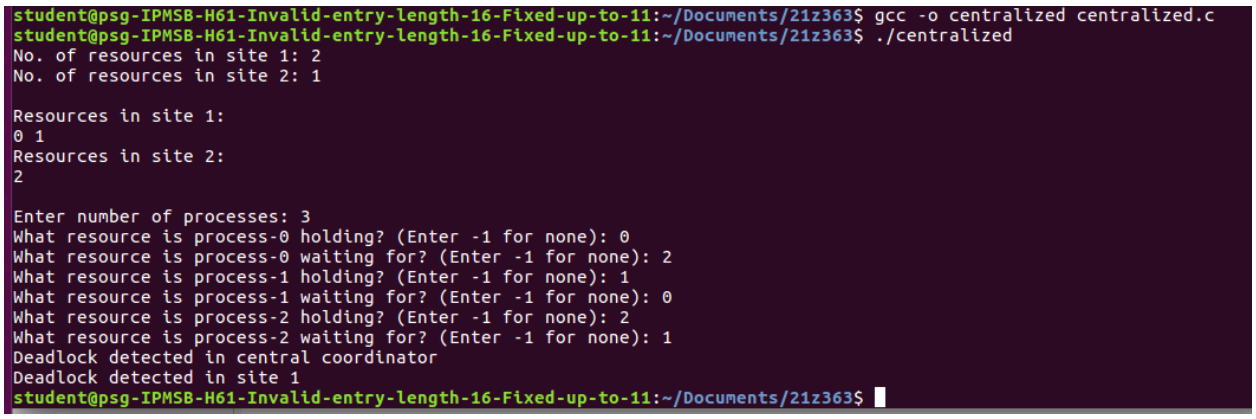
IMPLEMENTATION OF DEADLOCK DETECTION ALGORITHMS

3.a) Centralized Algorithm in C:

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

#define MAX_RESOURCES 100
#define MAX_PROCESSES 100

// Forward declaration of Resource structure
typedef struct Resource Resource;
// Structure to represent a process
typedef struct {
    int id;
    Resource* holding;
    Resource* waiting;
} Process;
// Structure to represent a resource
struct Resource {
    int id;
    int site;
    int heldBy; // Process ID of the process holding this resource, -1 if not held
};

// Function to check for cycles in the resource allocation graph
bool detectCycle(Process* processes, Resource* resources, Process* cur, int start) {
    for (int i = 0; i < MAX_PROCESSES; i++) {
        if (cur->waiting != NULL && cur->waiting->id == processes[i].holding->id) {
            if (processes[i].id == start) {
                return true;
            } else {
                if (detectCycle(processes, resources, &processes[i], start)) {
                    return true;
                }
            }
        }
    }
    return false;
}
```

```
// Function to check for deadlock in a site
bool checkDeadlockSite(Process* processes, Resource* resources, int site) {
    for (int i = 0; i < MAX_PROCESSES; i++) {
        if (processes[i].id != -1 && processes[i].holding != NULL && processes[i].waiting != NULL
&& processes[i].holding->site == site && processes[i].waiting->site == site) {
            if (detectCycle(processes, resources, &processes[i], processes[i].id)) {
                return true;
            }
        }
    }
    return false;
}
// Function to check for deadlock in the coordinator
bool checkDeadlock(Process* processes, Resource* resources) {
    for (int i = 0; i < MAX_PROCESSES; i++) {
        if (processes[i].waiting != NULL && detectCycle(processes, resources, &processes[i],
processes[i].id)) {
            // Check if the waiting resource is from a different site
            bool waitingFromDifferentSite = false;
            for (int j = 0; j < MAX_PROCESSES; j++) {
                if (processes[j].id != -1 && processes[j].holding != NULL && processes[j].waiting !=
NULL) {
                    if (processes[j].holding->site != processes[j].waiting->site) {
                        waitingFromDifferentSite = true;
                        break;
                    }
                }
            }
            if (waitingFromDifferentSite) {
                return true; // Global deadlock detected
            } else {
                return false; // Deadlock within a site, not global
            }
        }
    }
    return false;
}
int main() {
    Resource resources[MAX_RESOURCES];
    Process processes[MAX_PROCESSES];
    // Initialize processes
    for (int i = 0; i < MAX_PROCESSES; i++) {
        processes[i].id = -1; // Indicates empty slot
        processes[i].holding = NULL;
        processes[i].waiting = NULL;
    }

    // Initialize resources for site 1
    int s1No, s2No;
    printf("No. of resources in site 1: ");
    scanf("%d", &s1No);
    for (int i = 0; i < s1No; i++) {
        resources[i].id = i;
        resources[i].site = 1;
        resources[i].heldBy = -1; // Initially not held by any process
    }

    // Initialize resources for site 2
    printf("No. of resources in site 2: ");
    scanf("%d", &s2No);
    for (int i = s1No; i < s1No + s2No; i++) {
        resources[i].id = i;
        resources[i].site = 2;
        resources[i].heldBy = -1; // Initially not held by any process
    }

    printf("\nResources in site 1:\n");
    for (int i = 0; i < s1No; i++) {
        printf("%d ", resources[i].id);
    }
    printf("\nResources in site 2:\n");
    for (int i = s1No; i < s1No + s2No; i++) {
        printf("%d ", resources[i].id);
    }
    printf("\n\n");

    // Input processes
    int NoOfProcesses;
    printf("Enter number of processes: ");
    scanf("%d", &NoOfProcesses);
    for (int i = 0; i < NoOfProcesses; i++) {
        int hld, wai;
        printf("What resource is process-%d holding? (Enter -1 for none): ", i);
        scanf("%d", &hld);
        printf("What resource is process-%d waiting for? (Enter -1 for none): ", i);
        scanf("%d", &wai);
        processes[i].id = i;
        if (hld != -1) {
            processes[i].holding = &resources[hld];
            resources[hld].heldBy = i; // Process i is holding resource hld
        } else {
            processes[i].holding = NULL;
        }
        if (wai != -1) {
            processes[i].waiting = &resources[wai];
        } else {
            processes[i].waiting = NULL;
        }
    }

    bool globalDeadlock = checkDeadlock(processes, resources);
    bool site1Deadlock = checkDeadlockSite(processes, resources, 1);
    bool site2Deadlock = checkDeadlockSite(processes, resources, 2);

    if (globalDeadlock) {
        printf("Deadlock detected in central coordinator\n");
    }
    if (site1Deadlock) {
        printf("Deadlock detected in site 1\n");
    }
    if (site2Deadlock) {
        printf("Deadlock detected in site 2\n");
    }
    if (!globalDeadlock && !site1Deadlock && !site2Deadlock) {
        printf("No deadlock detected\n");
    }

    return 0;
}
```

OUTPUT:



```
student@psg-IPMSB-H61-Invalid-entry-length-16-Fixed-up-to-11:~/Documents/21z363$ gcc -o centralized centralized.c
student@psg-IPMSB-H61-Invalid-entry-length-16-Fixed-up-to-11:~/Documents/21z363$ ./centralized
No. of resources in site 1: 2
No. of resources in site 2: 1

Resources in site 1:
0 1
Resources in site 2:
2

Enter number of processes: 3
What resource is process-0 holding? (Enter -1 for none): 0
What resource is process-0 waiting for? (Enter -1 for none): 2
What resource is process-1 holding? (Enter -1 for none): 1
What resource is process-1 waiting for? (Enter -1 for none): 0
What resource is process-2 holding? (Enter -1 for none): 2
What resource is process-2 waiting for? (Enter -1 for none): 1
Deadlock detected in central coordinator
Deadlock detected in site 1
student@psg-IPMSB-H61-Invalid-entry-length-16-Fixed-up-to-11:~/Documents/21z363$
```