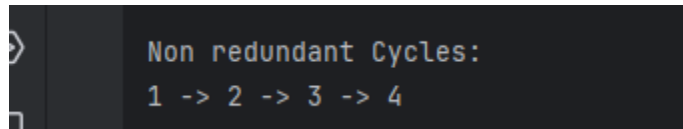


1174066

Deadlock Management Modules Implementation (graph reduction algorithm and deadlock detection algorithm)

Implementation final output

A terminal window with a dark background and light-colored text. It displays the output of a program. The first line is "Non redundant Cycles:" and the second line is "1 -> 2 -> 3 -> 4".

```
> Non redundant Cycles:  
1 -> 2 -> 3 -> 4
```

The implementation includes A graph reduction algorithm that reduces process-resource allocation graphs into process dependency graphs and A deadlock detection algorithm, together they work to output non redundant cycle of process dependency graph.

The Process class manages process-resource allocation, storing information like process number, allotted resource, and requested resource. The Reduction class handles dependency reduction, parsing data to create process objects, establishing dependencies based on resource allocation, transforming processes with dependencies into tuples, and printing dependency information.

The Cycle class in another part of the codebase focuses on cycle detection and processing in a directed graph, using depth-first search (DFS) for cycle detection, creating a graph from dependencies, processing cycles with and without the last number, and checking cycle similarity. The two classes are utilized together in managing and analyzing resource allocation and dependency relationships in a system.

Detailed output

Dependency Matrix:

	P1	P2	P3	P4
P1	0	1	0	0
P2	0	0	1	0
P3	0	0	0	1
P4	1	0	0	0

List of Dependencies:

(1, 2)
(2, 3)
(3, 4)
(4, 1)

Cycles with last number:

1 -> 2 -> 3 -> 4 -> 1
2 -> 3 -> 4 -> 1 -> 2
3 -> 4 -> 1 -> 2 -> 3
4 -> 1 -> 2 -> 3 -> 4

Cycles without last number:

1 -> 2 -> 3 -> 4
2 -> 3 -> 4 -> 1
3 -> 4 -> 1 -> 2
4 -> 1 -> 2 -> 3

Non redundant Cycles:

1 -> 2 -> 3 -> 4

Process finished with exit code 0