CSCI3100 Software Engineering

Assignment 4 Solution

1. Software Design Principle and Approach

(1) (5 pts) TDN

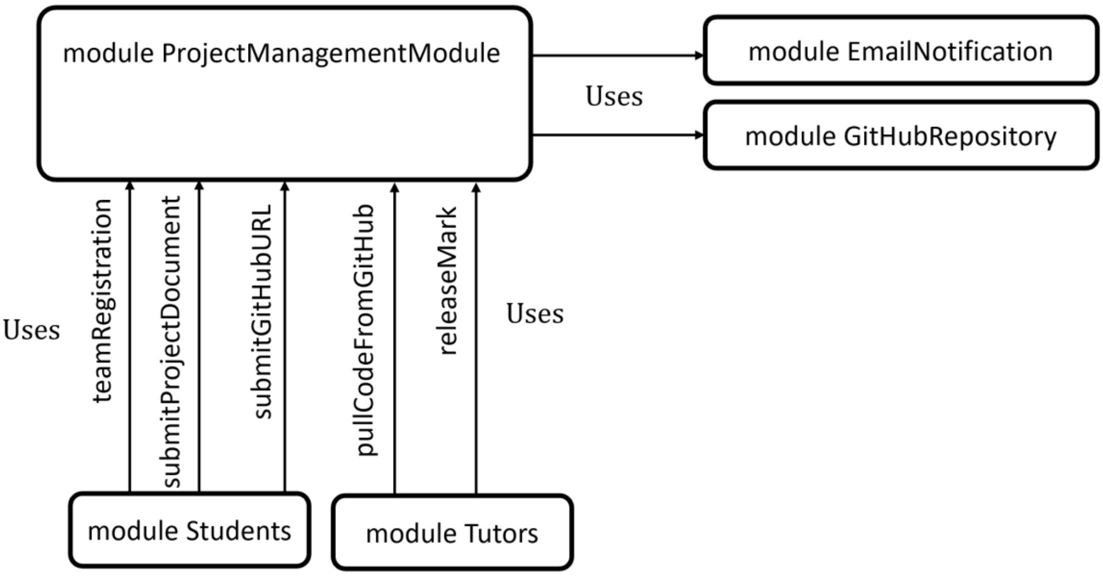
```
module GitHubRepository
exports
   type REPOSITORY: ?;
     This is an abstract data type module for GitHub repository;
the data structure is a secret hidden in the implementation part.
   procedure pullCode(url: in STRING, repo: in out REPOSITORY)
   This procedure pulls code from the given `url` and store the code
in `repo`.
   function generateReport(repo: in REPOSITORY): STRING
   This procedure generates the report of `repo` and returns a
STRING status report.
end GitHubRepository
```

(2) (14 pts) TDN

```
module ProjectManagementModule
uses
   EmailNotification, GitHubRepository
exports
     function teamRegistration(studentID: in ARRAY[1..5] of INT,
studentName: in ARRAY[1..5] of STRING, projectName: IN STRING): INT
     This function returns the generated teamID.
     procedure submitGitHubURL(teamID: in INT, url: in STRING)
     procedure submitProjectDocument(teamID: in INT, type: in
STRING, doc: in FILE)
     procedure pullCodeFromGitHub(teamID: in INT)
     procedure releaseMark(teamID: in INT, type: in STRING, mark:
in FLOAT, comment: in STRING)
implementation
     is_composed_of
     function checkMemberConflict(studentID: in INT): BOOLEAN
     This function checks whether the student has been included in
```

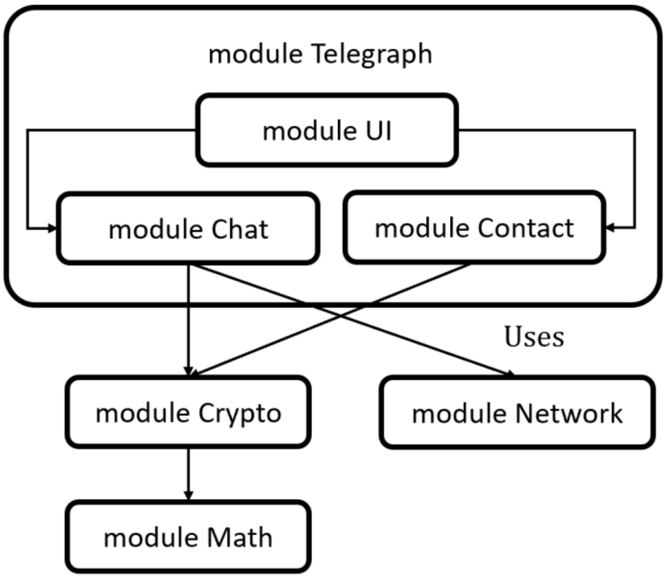
```
other teams. This function should not be exposed to outside.
end ProjectManagementModule
```

(3) (6 pts) GDN for the whole system (add “students” and “tutors” modules; all labels inside the modules should appear as resources, not inside the modules)

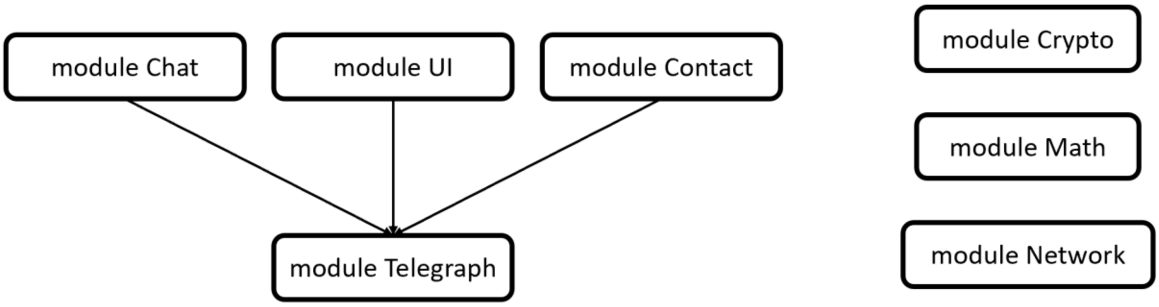


2. Software Design Principle and Approach

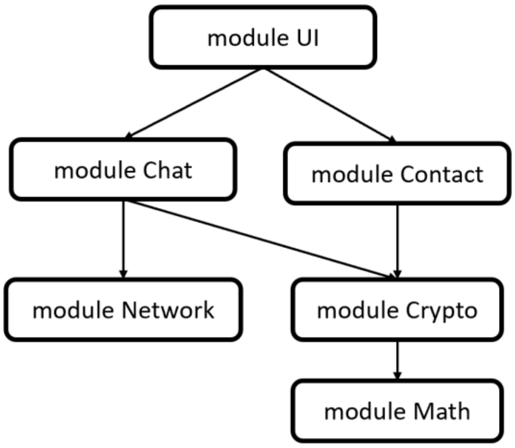
(1) (7 pts)



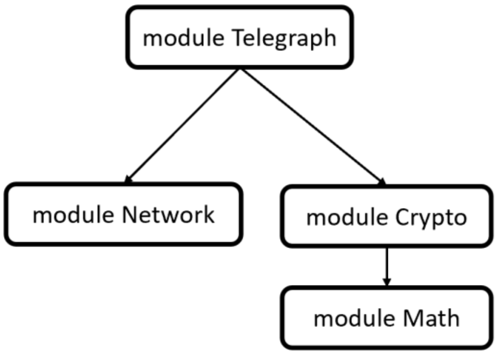
(2) (6 pts)



IS_COMPONENT_OF



USES



USES

(3) (7 pts)

Module	d^{in}	d^{out}	Stability (S)
Telegraph	0	3	0
UI	0	5	0
Chat	1	3	0.25
Contact	1	2	0.33
Network	2	0	1
Crypto	3	1	0.75
Math	5	0	1

(4) (5 pts)

“Stability” of a module is regarding the degree of being stable of the module in the whole system. When a module is stabilized in a system, it will have high stability measure and, as a result, it should not change often.

High “stability” means changing a module will require more potential changes to the clients of the used module. Low “stability” means changing a module requires fewer potential changes to the clients of the used module. For example, if the interface of *Crypto* is changed, then all its clients must be changed for correct operations.

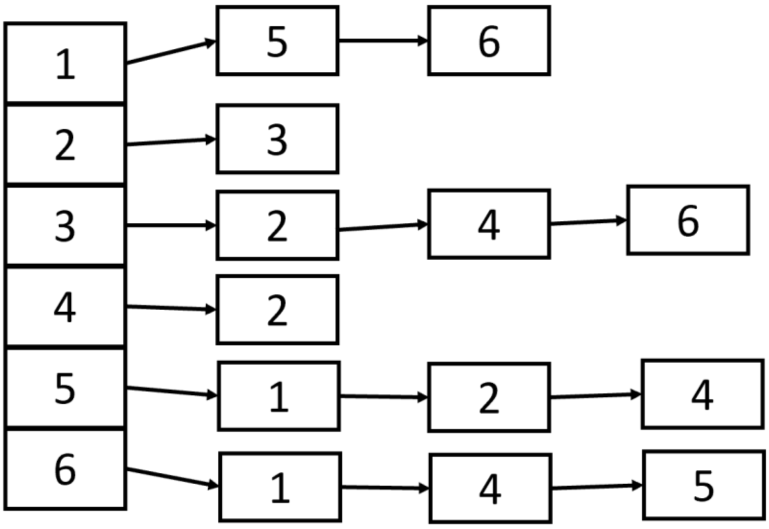
The “stability” measure encourages developers to:

- Build proper hierarchy relations: separate different modules by separating the responsibilities, i.e., “one module should focus on one thing.” The less responsibility a module have, the less likely it need to be changed.
- Decouple modules that changes occasionally and frequently: decouple a module if one part of the module changes frequently (e.g., the UI module) and another part changes less frequently (e.g., the network and math module).

3. Program Design Technique

Answer:

(1) (10 pts) Adjacency list



Adjacency Matrix

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

(2) (15 pts)

Solution 1: DFS $O(V * (V + E))$

Step 1

```
FOR each node in the graph
    use depth first traversal to check if there is ring
End FOR
```

Step 2

```
FOR node n in graph
    add node n to the VISITED array
    add every node in n's adjacency list to TO_VISIT queue
    WHILE TO_VISIT is not empty
        visit nodes in TO_VISIT
        if node in TO_VISIT is in the VISITED array then there is ring
    End WHILE
End FOR
```

Solution 2: Topological sort $O(V + E)$

Step 1

```
IF the out degree of all nodes are > 0 then then there is ring
IF the in degree of all nodes are > 0 then then there is ring
Apply topological sort from nodes with in degree == 0
```

Step 2

```
IF the out degree of all nodes are > 0 then then there is ring
IF the in degree of all nodes are > 0 then then there is ring
Put all nodes with in degree == 0 to the NEXT array
VISITED = empty array
WHILE NEXT is not empty
    delete a node u from NEXT and add u to the end of VISITED
    FOR every node v in n's adjacency list
        indegree of v minus 1
    End FOR
    Put all nodes with in degree == 0 to the NEXT array
END WHILE
```

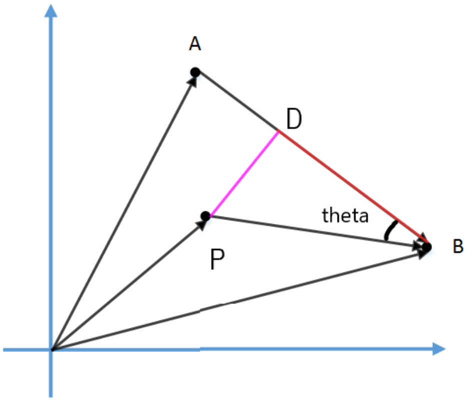
```
If not all nodes are visited then there is ring
```

NOTE: 5 pts for step 1, 8 pts for step 2, 2 pts for time complexity

4. Software Design: Recursive and Non-Recursive Modules

Answer:

(1) (5 pts)



$$\overrightarrow{AB} = \vec{B} - \vec{A}$$

$$\overrightarrow{PB} = \vec{B} - \vec{P}$$

$\overrightarrow{DB}$ is the projection of $\overrightarrow{PB}$ on $\overrightarrow{AB}$:

$$|\overrightarrow{BD}|^2 = \left(\frac{\overrightarrow{AB} \times \overrightarrow{PB}}{|\overrightarrow{AB}|}\right)^2$$

By the Pythagorean theorem we get:

$$|\overrightarrow{PD}| = \sqrt{|\overrightarrow{PB}|^2 - \frac{\left(\overrightarrow{AB} \times \overrightarrow{PB}\right)^2}{|\overrightarrow{AB}|^2}} = \sqrt{\left(y_p - y_b\right)^2 + \left(x_p - x_b\right)^2 - \frac{\left((x_b - x_a)(x_b - x_p) + (y_b - y_a)(y_b - y_p)\right)^2}{(y_b - y_a)^2 + (x_b - x_a)^2}}$$

NOTE1: You get 2 pts if you only describe how to calculate without giving the final result.

NOTE2: You get 2 pts if you give the final result.

NOTE3: There are many other ways to solve the question.

(2) (5 pts for each blank, 1 more pt if all blanks are correct)

① `pDistance`

② `points[idx]`

③ `points[0]`

④ `dist`

⑤ `maxDist`

(3) (5 pts for each blank, 1 more pt if all blanks are correct)

① `2`

② `maxDist >= threshold`

③ `points[0:maxIdx + 1]`

④ `points[maxIdx:]`

⑤ `False`

(4) (7 pts for each blank, 1 more pt if all blanks are correct)

① `len(points) – 1`

② `stack.empty()`

③ `e[0] + 1 == e[1]`

④ `e[0]`

⑤ `e[1] + 1`

⑥ `e[0]`

⑦ `maxIdx`