MIS Example

CS 2ME3/SE 2AA4

Sepehr Bayat

Department of Computing and Software
McMaster University

January 29

Outline

What is a module?

- A file with encapsulated code to implement a specific functionality
- Ex: For designing a website with a login system, we may have a module that deals with logging out
- A module comes with an "interface"
- An interface includes things like functions and arguments of the function
- Interface connects two system/modules, like a touch screen between people and phone
- Implementation and Interface are separated, no matter how the code changes, the output should always be the same

What is a MIS?

- Module Interface Specification
- Specifices externally observable behaviour of a module
- Not in language of implementation, uses mathematical language
- Implementation details are not included in the MIS
- MIS uses mathematical language for its preciseness, which means that it is helpful to learn discrete math from the last tutorial

MIS Template Structure

- Uses
  - Imported constants, data types and access programs
- Syntax
  - Exported constants and types
  - Exported functions (access routine interface syntax)
- Semantics
  - State variables
  - State invariants
  - Assumptions
  - Access routine semantics
  - Local functions
  - Local types
  - Local constants
  - Considerations

Example

Consider the implementation of a triangle on a 2-D plane

- A point is represented by pair of real numbers (x,y)
- A triangle is represented by three points
- The triangle should be able to determine that it is a valid triangle, calculate its perimeter and calculate its area.

We are using the following inequality to test that the triangle is valid:

$$\mathrm{AB} + \mathrm{AC} > \mathrm{BC}, \mathrm{AC} + \mathrm{BC} > \mathrm{AB}, \mathrm{AB} + \mathrm{BC} > \mathrm{AC}$$

We are calculating the perimeter of the triangle using the following formula:

$$\mathrm{P} = (\mathrm{AB} + \mathrm{AC} + \mathrm{BC})$$

We are using the Heron formula to calculate the area of the triangle:

$$\sqrt[2]{P/2(P/2 - AB)(P/2 - AC)(P/2 - BC)}$$

Point ADT Module

Template Module
PointADT

Uses
N/A

Syntax
Exported Types:
PointT = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new PointT	real, real	PointT	
xcoord		real	
ycoord		real	
dist	PointT	real	

Semantics

State Variables:

xc: real

yc: real

State Invariant None

Assumptions None

Point ADT Module

Access Routine Semantics

new PointT(x, y):

- transition: $xc, yc := x, y$
- output: $out := self$
- exception: none

xcoord():

- output: $out := xc$
- exception: none

ycoord():

- output: $out := yc$
- exception: none

dist(p):

- output: $out := \sqrt{(self.xc - p.\text{xcoord}())^2 + (self.yc - p.\text{ycoord}())^2}$
- exception: none

MIS Interface

From the MIS we can deduce the interface of the code will look like:

```
# Interface
class PointT:

    # Constructor
    def __init__(self, x, y):

    # Selectors
    def xcoord(self):

    def ycoord(self):

    def dist(self, p):
```

MIS Implementation

See PointADT.py for implementation.

TriangleADT

Template Module
TriangleADT

Uses
PointADT

Syntax
Exported Types:
TriangleT = ?

Exported Access Programs

Routine name	In	Out	Except.
new TriangleT	PointT, PointT, PointT	TriangleT	
sides		seq[3] of real	
inequality_theorem		boolean	LINE
perimeter_of_triangle		real	LINE
area_of_triangle		real	LINE

Semantics

State Variables:

$p1$: PointT

$p2$: PointT

$p3$: PointT

State Invariant None

Assumptions None

Access Routine Semantics

new TriangleT(a, b, c):

- transition: $p1, p2, p3 := a, b, c$
- output: $out := self$
- exception: none

sides():

- output: $out := [p1.dist(p2), p1.dist(p3), p2.dist(p3)]$
- exception: none

inequality_theorem():

- output: $out := ((self.sides[0] + self.sides[1]) > self.sides[2] \wedge (self.sides[1] + self.sides[2]) > self.sides[0] \wedge (self.sides[0] + self.sides[2]) > self.sides[1])$
- exception: $ex := ((p1.xcoord() == p2.xcoord() == p3.xcoord() \vee p1.ycoord() == p2.ycoord() == p3.ycoord()) \Rightarrow \text{LINE})$

perimeter_of_triangle():

- output:

$$out := self.sides[0] + self.sides[1] + self.sides[2]$$

- exception:
  ex := ((p1.xcoord()==p2.xcoord() ==p3.xcoord() $\vee$
  p1.ycoord()==p2.ycoord() ==p3.ycoord()) $\Rightarrow$ LINE)

area_of_triangle():

- output : $out :=$
  $\sqrt[2]{P/2(P/2 - self.sides[0])(P/2 - self.sides[1])(P/2 - self.sides[2])}$
- exception:
  ex := ((p1.xcoord()==p2.xcoord() ==p3.xcoord() $\vee$
  p1.ycoord()==p2.ycoord() ==p3.ycoord()) $\Rightarrow$ LINE)

Local Function

P := perimeter_of_triangle( )

MIS Interface

From the MIS we can deduce the interface of the code will look like:

```
# Interface
class TriangleADT:

    #Constructor
    def __init__(self, p1, p2, p3):


    #Selectors
    def sides(self):

    def inequality_theorem(self):

    def perimeter_of_triangle(self):

    def area_of_triangle(self):
```

MIS Implementation

See TriangleADT for implementation.

Implementation files

- Implementation files PointADT.py and TriangleADT.py can be found in the repo under Tutorial/T4/src