# EECS3311 Software Design (Fall 2020)

## Q&A - Exam

Thursday, December 17

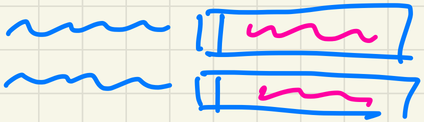
# Exam

2 hours

Sunday 9am

# Format

1. Multiple choice
   - single choice ☑ ○ ○
   - multle choices ☑ □ ☑
   - matching

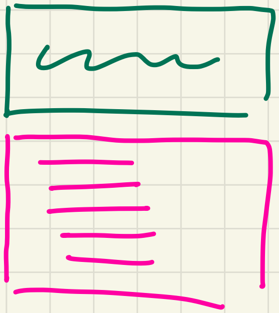


2. Short answer

→ ↳ not graded manually. e.g. single_choice_principle.

3. Essay questions.

→ graded manually.



≤ 5 sentences

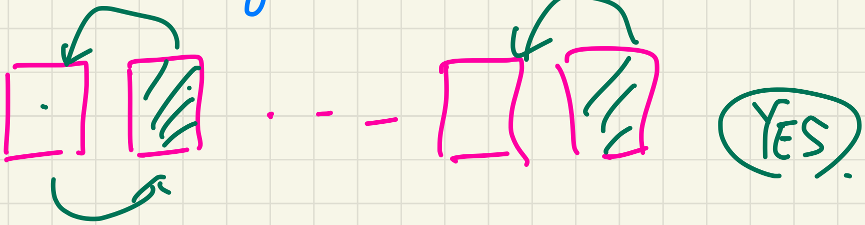
15 ~ 25

total marks : 200 190

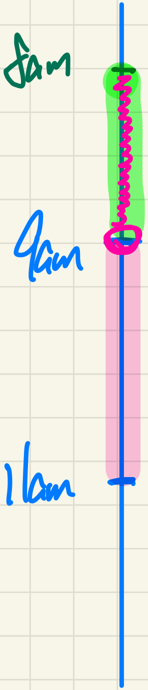
~~85%~~

95%.

- open book exam
- collaboration X
- Recall slide x bullet y ... X
- Each question is self contained



# Sunday

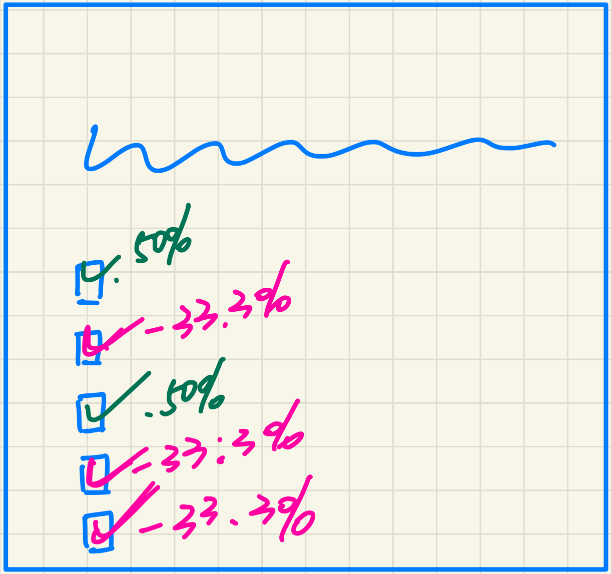


- start testing your network
- not responsive, get in touch

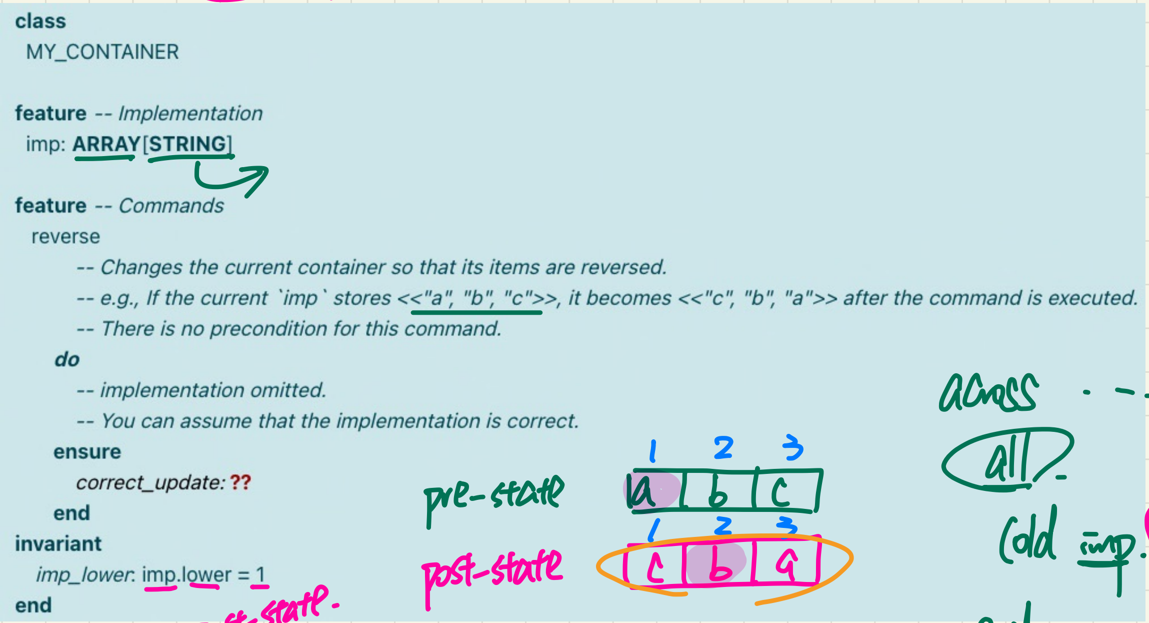
across a as i

a[~~i~~]

i. item



Ziwei: In quiz 3, can u explain these 2 please



```
class
  MY_CONTAINER

feature -- Implementation
  imp: ARRAY[STRING]

feature -- Commands
   reverse
        -- Changes the current container so that its items are reversed.
        -- e.g., If the current `imp` stores <<"a", "b", "c">>, it becomes <<"c", "b", "a">> after the command is executed.
        -- There is no precondition for this command.
     do
        -- implementation omitted.
        -- You can assume that the implementation is correct.
     ensure
        correct_update: ??
     end
invariant
   imp_lower: imp.lower = 1
end
```

pre-state

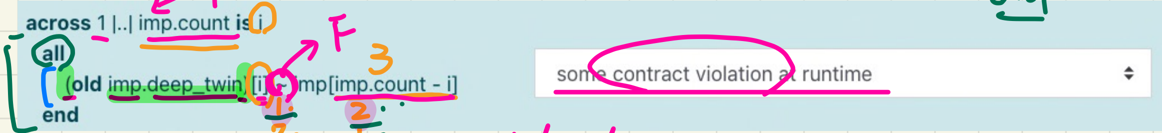
| 1 | 2 | 3 |
|---|---|---|
| a | b | c |

post-state

| 1 | 2 | 3 |
|---|---|---|
| c | b | a |

across ...
all
(old imp.d_t)[i] =
imp[imp.c-i+1]
end

post-state.



```
across 1 |..| imp.count is i
  all
    (old imp.deep_twin)[i] ~ imp[imp.count - i]
  end
```

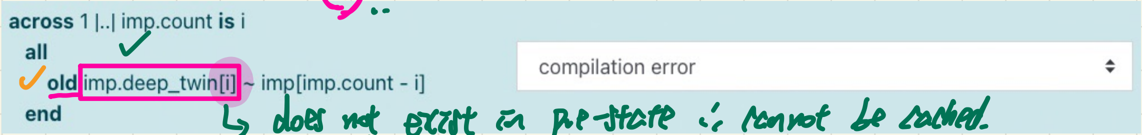
some contract violation at runtime

F 3

1. 2. 3.

2 1 .

0 → invalid index



```
across 1 |..| imp.count is i
  all
    old imp.deep_twin[i] ~ imp[imp.count - i]
  end
```

compilation error

↳ does not exist in pre-state ∴ cannot be cached.

P(1) ∧ P(2) ∧ P(3) ∧ ...

F

==

T

across [ ] is i
all
end [P(i)]

✓ compiles

of strings

F.

until [ ]

across [ ] ..
loop → no early exit.
end

# Recommended Exercises

1. Study Group ]

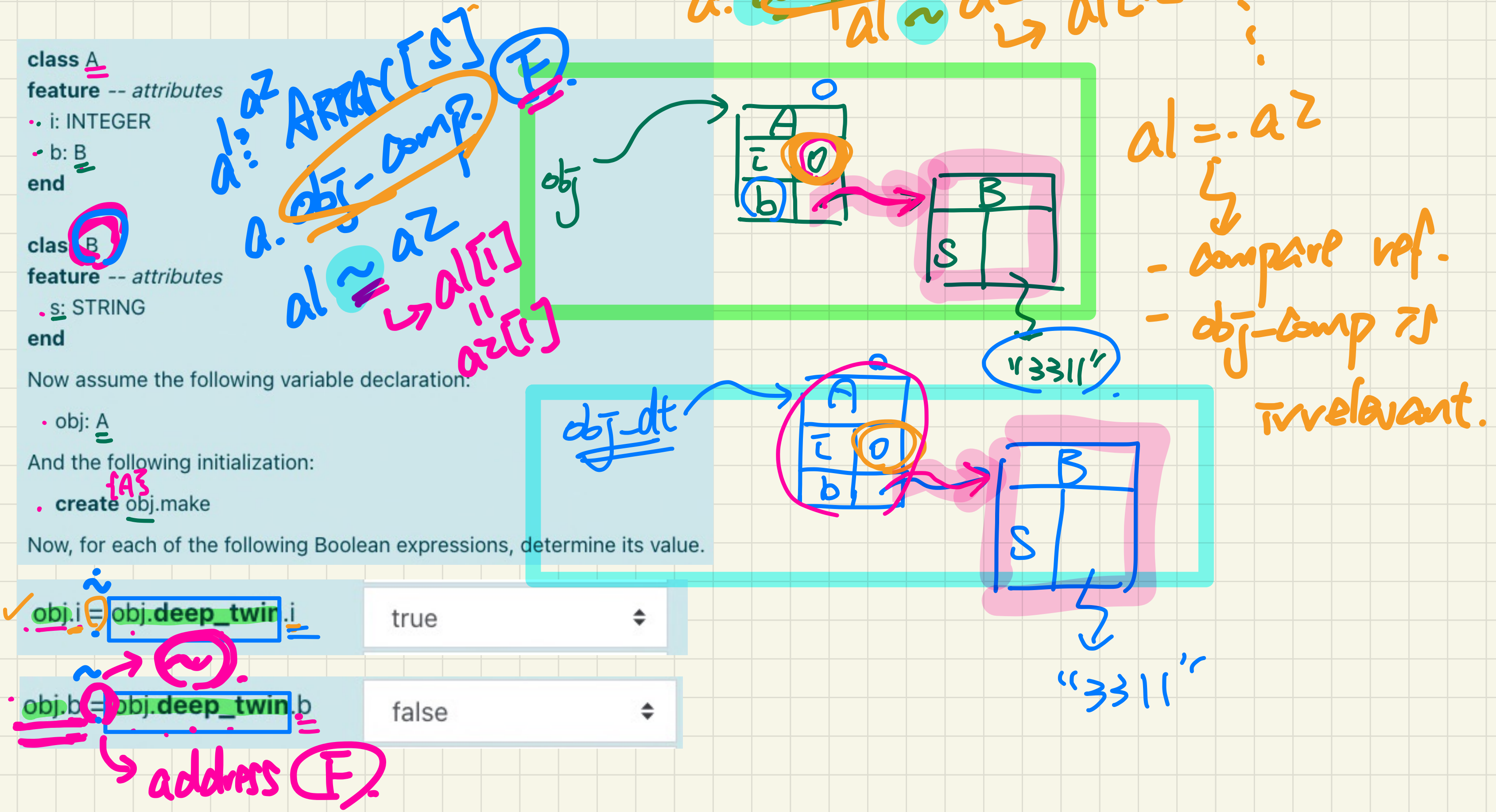
2. Go over quiz questions.

↳ Turn T/F or M.C. questions into short answers.

① Explain why the correct answer is the case.

② Explain why the incorrect answers are <u>not</u> the case.

Zhao: In quiz 2, why obj.i = obj.deep_twin.i is True, but obj.b = obj.deep_twin.b is False?



class A
feature -- *attributes*
i: INTEGER
b: B
end

class B
feature -- *attributes*
s: STRING
end

Now assume the following variable declaration:

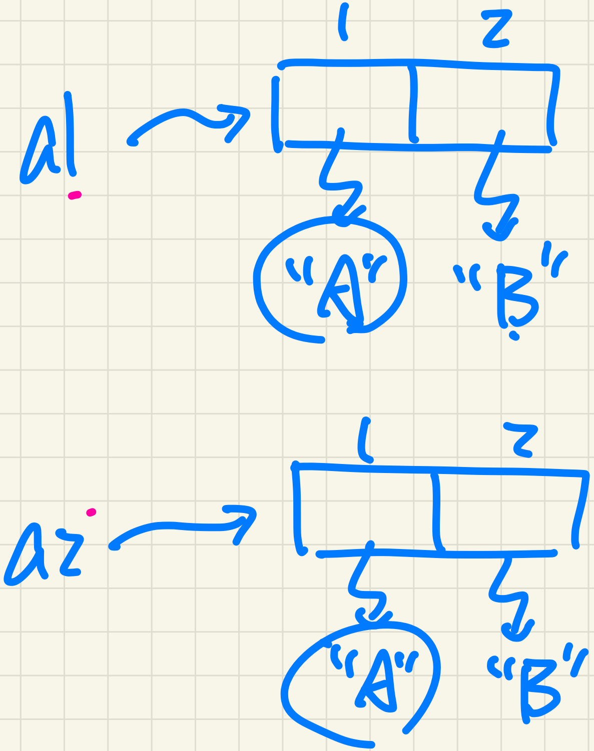
obj: A

And the following initialization:

create obj.make

Now, for each of the following Boolean expressions, determine its value.

| | |
|---|---|
| obj.i = obj.deep_twin.i | true |
| obj.b = obj.deep_twin.b | false |



a1.obj_comp.
a2.obj_comp. ] F.

a1 = a2 (F).

a1 ~ a2 (F).

↳ a1[1] = a2[1] (F).

a1.compare_obj

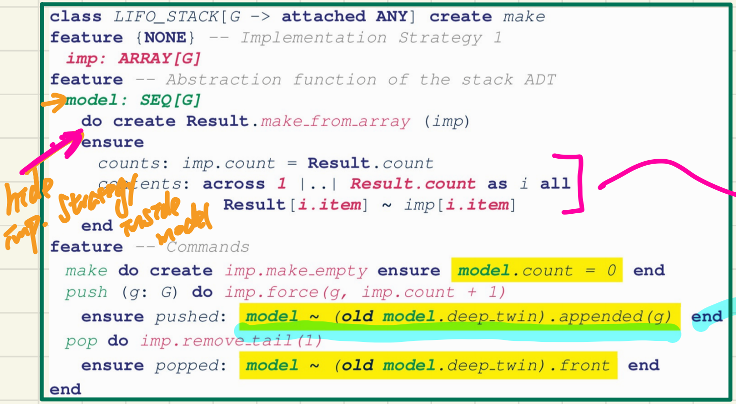
a2.compare_obj. is-equal fn STRING

a1 = a2 (F). (T).

a1 ~ a2 → a1[1] ~ a2[1]

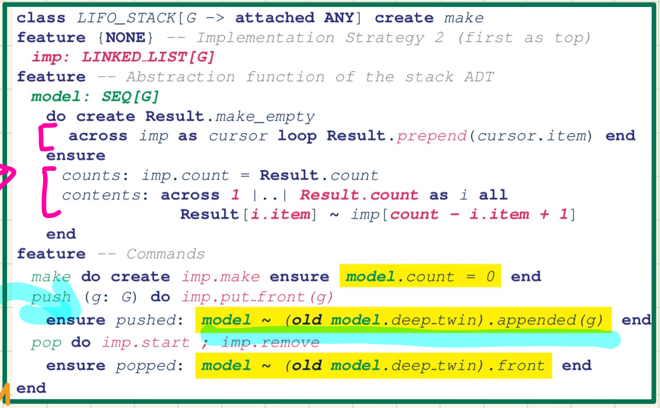
Amir: (Lecture 3b Part 3 Slide 17 of 35)

Instead of using model functions, cannot we use ARRAY as implementation and use some exported queries to implement the contracts of pre/post condition



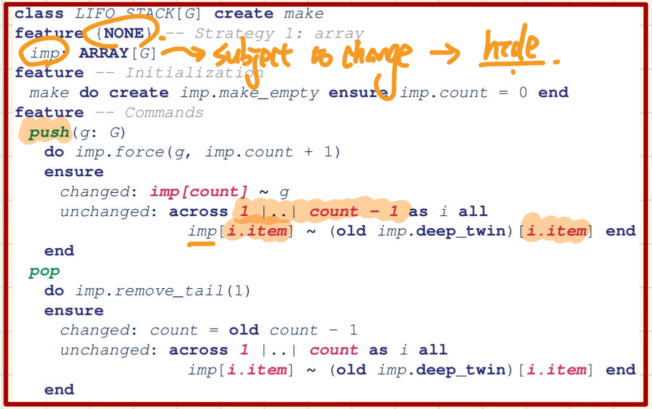
```
class LIFO_STACK[G -> attached ANY] create make
feature {NONE} -- Implementation Strategy 1
  imp: ARRAY[G]
feature -- Abstraction function of the stack ADT
  model: SEQ[G]
    do create Result.make_from_array (imp)
    ensure
      counts: imp.count = Result.count
      contents: across 1 |..| Result.count as i all
                  Result[i.item] ~ imp[i.item]
    end
feature -- Commands
  make do create imp.make_empty ensure model.count = 0 end
  push (g: G) do imp.force(g, imp.count + 1)
    ensure pushed: model ~ (old model.deep_twin).appended(g) end
  pop do imp.remove_tail(1)
    ensure popped: model ~ (old model.deep_twin).front end
end
```

hide Imp. Strategy Inside model



```
class LIFO_STACK[G -> attached ANY] create make
feature {NONE} -- Implementation Strategy 2 (first as top)
  imp: LINKED_LIST[G]
feature -- Abstraction function of the stack ADT
  model: SEQ[G]
    do create Result.make_empty
      across imp as cursor loop Result.prepend(cursor.item) end
    ensure
      counts: imp.count = Result.count
      contents: across 1 |..| Result.count as i all
                  Result[i.item] ~ imp[count - i.item + 1]
    end
feature -- Commands
  make do create imp.make ensure model.count = 0 end
  push (g: G) do imp.put_front(g)
    ensure pushed: model ~ (old model.deep_twin).appended(g) end
  pop do imp.start ; imp.remove
    ensure popped: model ~ (old model.deep_twin).front end
end
```

Information Hiding



```
class LIFO_STACK[G] create make
feature {NONE} -- Strategy 1: array
  imp: ARRAY[G]
feature -- Initialization
  make do create imp.make_empty ensure imp.count = 0 end
feature -- Commands
  push(g: G)
    do imp.force(g, imp.count + 1)
    ensure
      changed: imp[count] ~ g
      unchanged: across 1 |..| count - 1 as i all
                   imp[i.item] ~ (old imp.deep_twin)[i.item] end
    end
  pop
    do imp.remove_tail(1)
    ensure
      changed: count = old count - 1
      unchanged: across 1 |..| count as i all
                   imp[i.item] ~ (old imp.deep_twin)[i.item] end
    end
```

→ subject to change → hide.

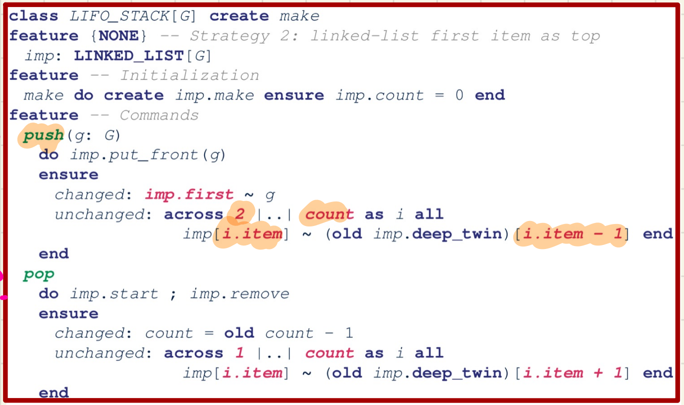
Imp change

⇓

Contract change.

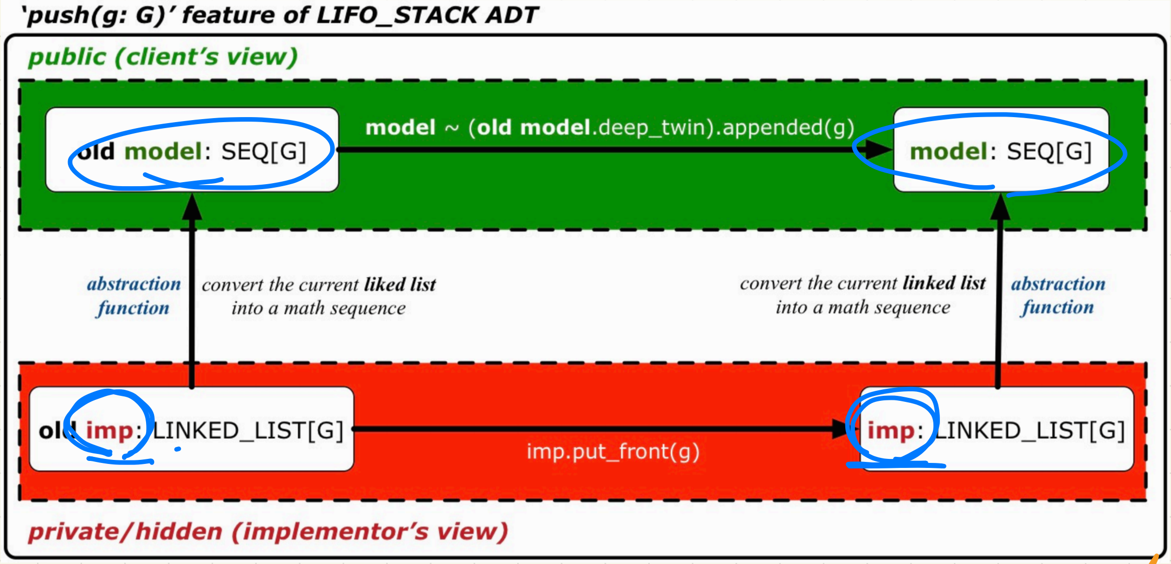
⇓

SCP X



```
class LIFO_STACK[G] create make
feature {NONE} -- Strategy 2: linked-list first item as top
  imp: LINKED_LIST[G]
feature -- Initialization
  make do create imp.make ensure imp.count = 0 end
feature -- Commands
  push(g: G)
    do imp.put_front(g)
    ensure
      changed: imp.first ~ g
      unchanged: across 2 |..| count as i all
                   imp[i.item] ~ (old imp.deep_twin)[i.item - 1] end
    end
  pop
    do imp.start ; imp.remove
    ensure
      changed: count = old count - 1
      unchanged: across 1 |..| count as i all
                   imp[i.item] ~ (old imp.deep_twin)[i.item + 1] end
    end
```

Amir: (Lecture 3b Part 2 - abstraction) -
this question is not directly related to the course.
For large amount of data (thousands of records) is it practical to have
model independent of data structure for information hiding?
It not is slow? Is it practical?



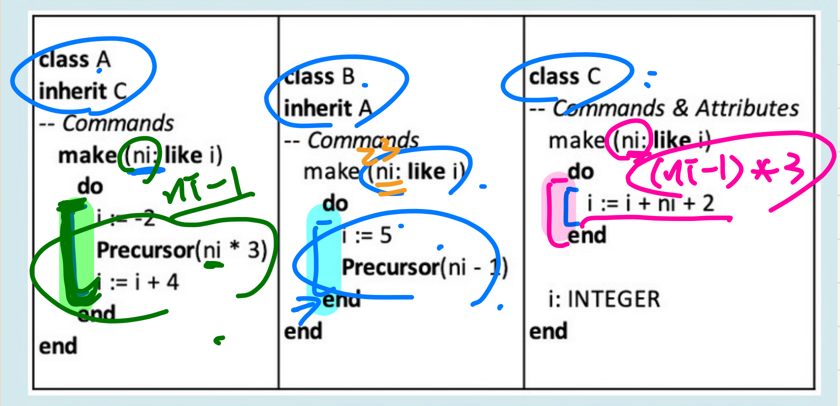
**'push(g: G)' feature of LIFO_STACK ADT**

*convert the current liked list into a math sequence*

*convert the current linked list into a math sequence*

- Critical parts of System (e.g., splay tree) → math model.
- Non-routine, algorithmically complex, operations

Cedric: Professor, please do you mind explaining in the detail the process one should use to get the answers to questions 4 and 5 of quiz 6 as shown in the images below:

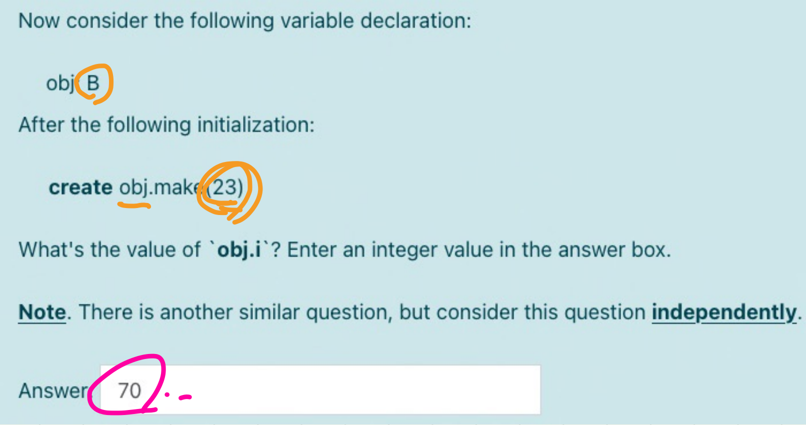


```
class A
inherit C
-- Commands
  make (ni: like i)
    do
      i := -2
      Precursor(ni * 3)
      i := i + 4
    end
end
```

```
class B
inherit A
-- Commands
  make (ni: like i)
    do
      i := 5
      Precursor(ni - 1)
    end
end
```

```
class C
-- Commands & Attributes
  make (ni: like i)
    do
      i := i + ni + 2
    end

  i: INTEGER
end
```



Now consider the following variable declaration:

obj: B

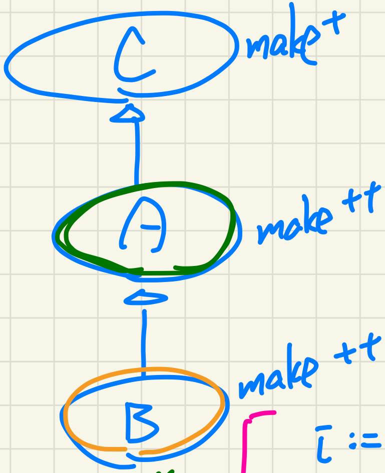
After the following initialization:

**create** obj.make(23)

What's the value of `` `obj.i` ``? Enter an integer value in the answer box.

**Note**. There is another similar question, but consider this question **independently**.

Answer: 70

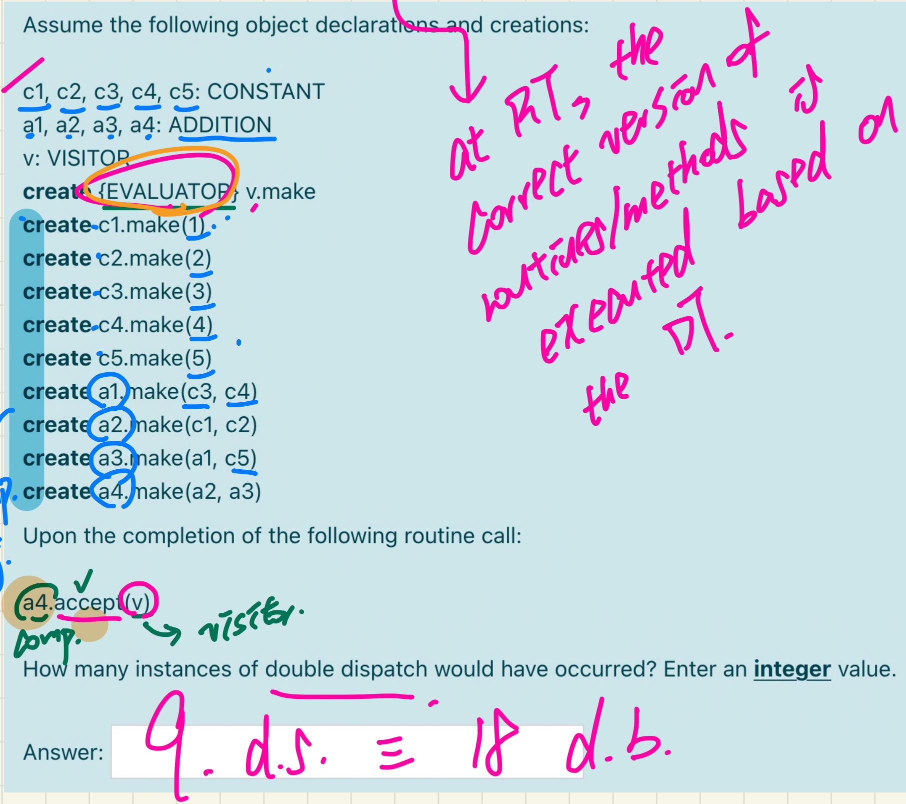


"Flattened" view of B's imp.

```
i := 5
~~Precursor(ni-1)~~
i := -2
~~Precursor((ni-1)*3)~~
i := i + (ni-1)*3 + 2
i := i + 4
```

Can you please tell us the difference between an instance of double dispatch and dynamic binding, also how to derive it.
Related to week 11 quiz, question 3 and 10. - mim

D.B. happens twice.



Assume the following object declarations and creations:

```
c1, c2, c3, c4, c5: CONSTANT
a1, a2, a3, a4: ADDITION
v: VISITOR
create {EVALUATOR} v.make
create c1.make(1)
create c2.make(2)
create c3.make(3)
create c4.make(4)
create c5.make(5)
create a1.make(c3, c4)
create a2.make(c1, c2)
create a3.make(a1, c5)
create a4.make(a2, a3)
```

comp. obj.

Upon the completion of the following routine call:

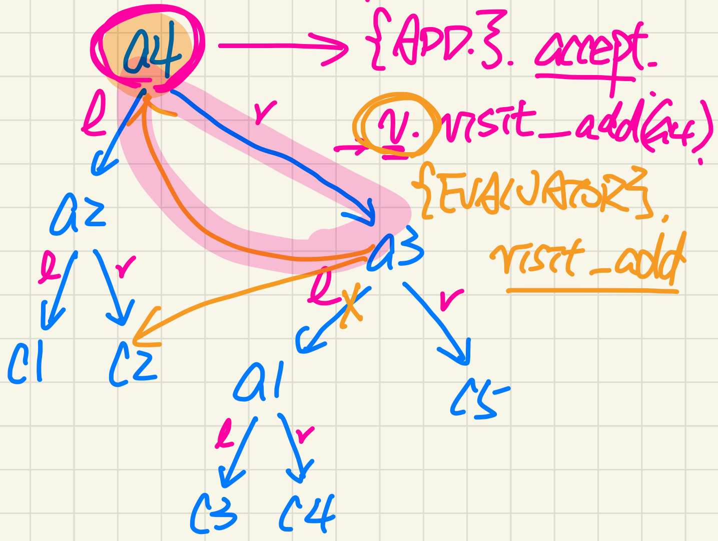
```
a4.accept(v)
```

comp. visitor.

How many instances of double dispatch would have occurred? Enter an **integer** value.

Answer: 9. d.s. ≡ 18 d.b.

at RT, the correct version of routines/methods is executed based on the DT.

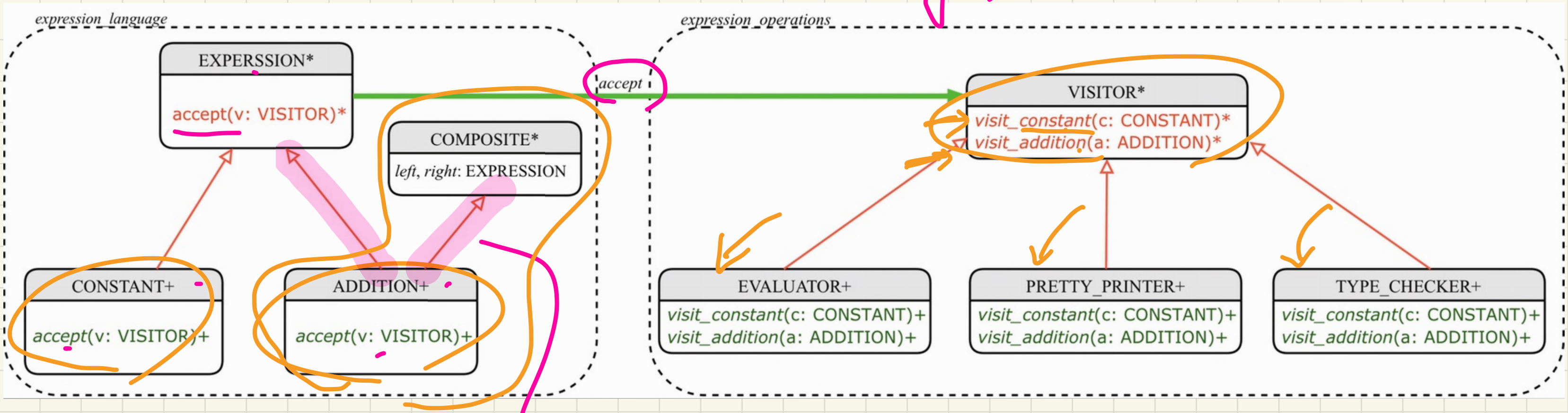


a3.make(a4, c2)

Amir:

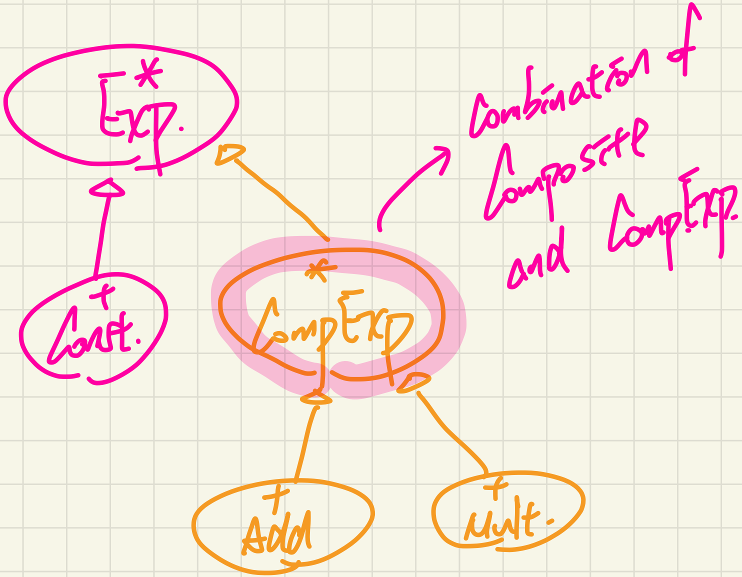
( W11 - visitor model, this question is not directly related to the course) : visitor model is based on multiple inheritance. In languages like Java which does not offer true multiple inheritance, there is no visitor pattern?

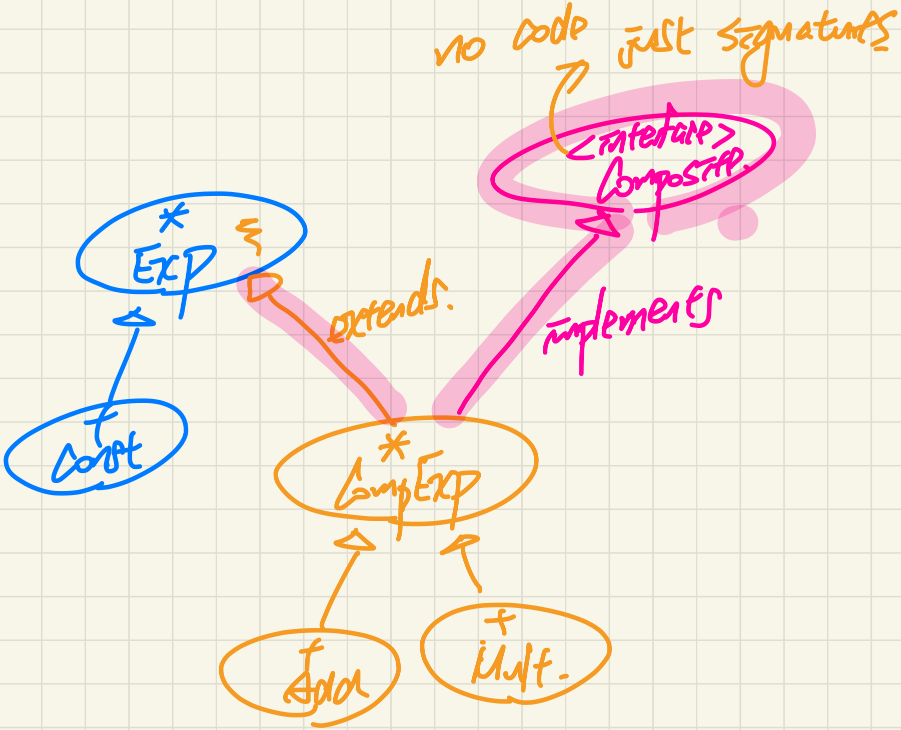
yes, there's visitor pattern.


expression_language
EXPERSSION*
accept(v: VISITOR)*
COMPOSITE*
left, right: EXPRESSION
CONSTANT+
accept(v: VISITOR)+
ADDITION+
accept(v: VISITOR)+
accept
expression_operations
VISITOR*
visit_constant(c: CONSTANT)*
visit_addition(a: ADDITION)*
EVALUATOR+
visit_constant(c: CONSTANT)+
visit_addition(a: ADDITION)+
PRETTY_PRINTER+
visit_constant(c: CONSTANT)+
visit_addition(a: ADDITION)+
TYPE_CHECKER+
visit_constant(c: CONSTANT)+
visit_addition(a: ADDITION)+

multiple inheritance.

# Java.


Exp. *
Const. +
CompExp *
Add +
Mult. +
Combination of Composite and CompExp


no code just signatures
<interface> Composite.
Exp * ≡
Const +
extends.
implements
CompExp *
Add +
Mult. +