	class RESIDENT_STUDENT
1st Design Attempt	create make
131 Design Arreinpr	feature Attributes
	name: STRING
	courses: LINKED_LIST[COURSE]
Good decign?	premium_rate: REAL
Good design?	feature Constructor
	make (n: STRING)
Judge by Cohesion	do name := n ; create courses.make end
	feature Commands
	<pre>set_pr (r: REAL) do premium_rate := r end</pre>
	register (c: COURSE) do courses.extend (c) end
class NON_RESIDENT_STUDENT	feature Queries
create make	tuition: REAL
feature Attributes	local base: REAL
name: STRING	do base := 0.0
	across courses as c loop base := base + c.item.fee end
courses: LINKED_LIST[COURSE]	Result := base * premium_rate
discount_rate: REAL	end
feature Constructor	
make (n: STRING)	end
de name := n : greate courses m	a lea and

do name := n ; create courses.make end
feature -- Commands

set_dr (r: REAL) do discount_rate := r end

register (c: COURSE) do courses.extend (c) end
feature -- Oueries

tuition: REAL

local base: REAL
do base := 0.0

across courses as c loop base := base + c.item.fee end

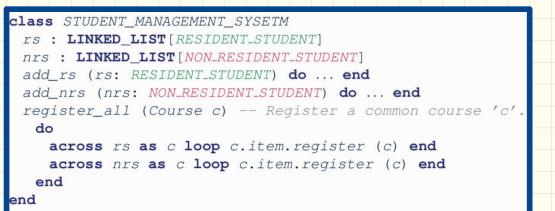
Result := base * discount_rate

end end

```
class RESIDENT_STUDENT
1st Design Attempt
                                     create make
                                     feature -- Attributes
                                      name: STRING
                                      courses: LINKED LIST [COURSE]
 Good design?
                                       premium_rate: REAL
 Judge by Single Choice Principle
                                     feature -- Constructor
                                      make (n: STRING)
 - A new kind is introduced?
                                        do name := n ; create courses.make end
                                     feature -- Commands
 - Change on registration policy?
                                       set_pr (r: REAL) do premium_rate := r end
                                       register (c: COURSE) do courses.extend (c) end
                                     feature -- Oueries
 class NON RESIDENT STUDENT
                                      tuition: REAL
 create make
                                        local base: REAL
 feature -- Attributes
                                        do base := 0.0
  name: STRING
                                           across courses as c loop base := base + c.item.fee end
   courses: LINKED LIST[COURSE]
                                           Result := base * premium_rate
   discount_rate: REAL
                                        end
 feature -- Constructor
                                     end
  make (n: STRING)
    do name := n ; create courses.make end
 feature -- Commands
   set_dr (r: REAL) do discount_rate := r end
   register (c: COURSE) do courses.extend (c) end
 feature -- Oueries
   tuition: REAL
    local base: REAL
    do base := 0.0
       across courses as c loop base := base + c.item.fee end
       Result := base * discount_rate
    end
 end
```

	class RESIDENT_STUDENT	
1st Design Attempt	create make feature Attributes	
	name: STRING	
	courses: LINKED_LIST[COURSE]	
Good design?		
	premium_rate: REAL	
How do you build a	feature Constructor	
STUDENT_MANGEMENT_SYSTEM	<pre>make (n: STRING) do name := n ; create courses.make end</pre>	
class accordingly?	feature Commands	
class accordingly?		
	set_pr (r: REAL) do premium_rate := r end	
	register (c: COURSE) do courses.extend (c) end feature Oueries	
class NON_RESIDENT_STUDENT	tuition: REAL	
create make	local base: REAL	
feature Attributes	do base := 0.0	
name: STRING	across courses as c loop base := base + c.item.fee end	
courses: LINKED_LIST[COURSE]	Result := base * premium_rate	
discount_rate: REAL	end	
feature Constructor	end	
make (n: STRING)		
do name := n ; create courses.make end feature Commands		
set_dr (r: REAL) do discount_rate := r end		
register (c: COURSE) do courses.extend (c) end		
feature Queries tuition: REAL		
local base: REAL		
do base := 0.0		
across courses as c loop base := base + c.item.fee end		
Result := base * discount_rate end		
end		

Without Inheritance (Design 1) Collection of Students



<u>Clinet's Code</u>

c: COURSE rs: RESIDENT_STUDENT nrs: NON_RESIDENT_STUDENT sms: SMS create c.make("3311") create sms.make

sms.add_rs(rs)
sms.add_nrs(nrs)
sms.register_all(c)

Q: What if more kinds of students are to be introduced?

2nd Design Attempt

class
STUDENT
create
make
feature attribures
courses: LINKED_LIST[COURSE
kind: INTEGER
premiumRate: REAL
discountRate: REAL
feature command
make (kind: INTEGER)
do
kind := a_kind
end
end

<u>Good design</u>? Judge by <mark>Cohesion</mark>

```
get_tuition: REAL
  local
   tuition: REAL
 do
    across courses is c loop
      tuition := tuition + c.fee
    end
    if kind = 1 then
      Result := tuition * premiumRate
    elseif kind = 2 then
      Result := tuition * discountRate
    end
  end
```

```
register (c: COURSE)
local
max: INTEGER
do
if kind = 1 then MAX := 6
elseif kind = 2 then MAX := 4
end
if courses.count = MAX then -- Error
else courses.extend (c)
end
end
```

2nd Design Attempt

STUDENT create make feature attribures courses: LINKED_LIST[COURSE] kind: INTEGER premiumRate: REAL discountRate: REAL
make feature attribures courses: LINKED_LIST [COURSE kind: INTEGER premiumRate: REAL
feature attribures courses: LINKED_LIST[COURSE kind: INTEGER premiumRate: REAL
courses: LINKED_LIST[COURSE kind: INTEGER premiumRate: REAL
kind: INTEGER premiumRate: REAL
premiumRate: REAL
discountRate: REAL
feature command
make (kind: INTEGER)
do
kind := a_kind
end
end

Good design?

Judge by Single Choice Principle

- A new kind is introduced?
- An existing kind is obeselete?

get_tuition: REAL
local
tuition: REAL
do
across courses is c loop
tuition := tuition + c.fee
end
if kind = 1 then
Result := tuition * premiumRate
elseif kind = 2 then
Result := tuition * discountRate
end
end

```
register (c: COURSE)

local

max: INTEGER

do

if kind = 1 then MAX := 6

elseif kind = 2 then MAX := 4

end

if courses.count = MAX then -- Error

else courses.extend (c)

end

end
```

2nd Design Attempt

class
STUDENT
create
make
feature attribures
courses: LINKED_LIST[COURSE
kind: INTEGER
premiumRate: REAL
discountRate: REAL
feature command
make (kind: INTEGER)
do
kind := a_kind
end
end

Good design?

How do you build a **STUDENT_MANGEMENT_SYSTEM** class accordingly?

```
get_tuition: REAL
  local
   tuition: REAL
 do
    across courses is c loop
      tuition := tuition + c.fee
    end
    if kind = 1 then
      Result := tuition * premiumRate
    elseif kind = 2 then
      Result := tuition * discountRate
    end
  end
```

```
register (c: COURSE)

local

max: INTEGER

do

if kind = 1 then MAX := 6

elseif kind = 2 then MAX := 4

end

if courses.count = MAX then -- Error

else courses.extend (c)

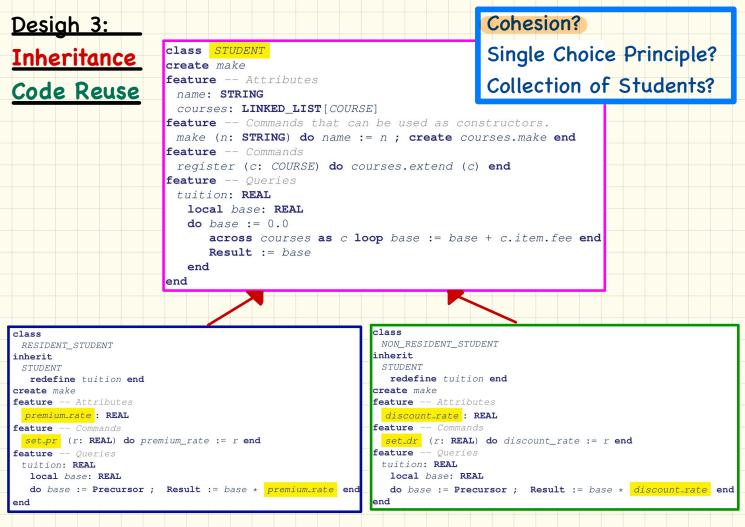
end

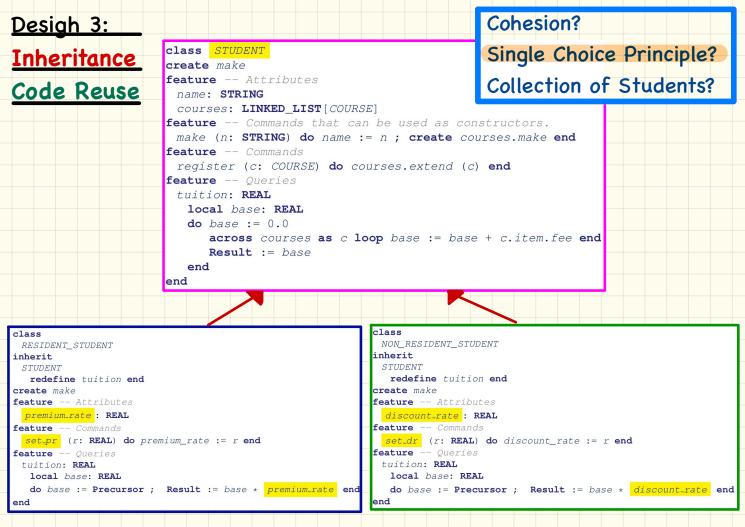
end
```

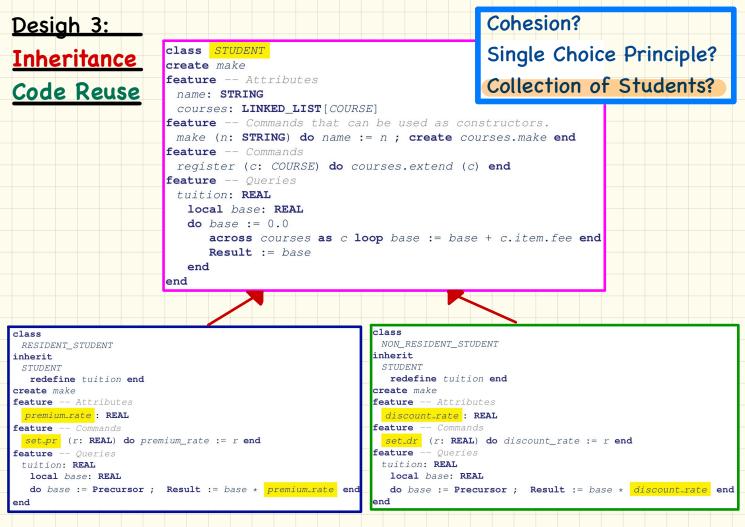
Without Inheritance (Design 2) Collection of Students

```
Clinet's Code
class
 STUDENT MANAGEMENT SYSTEM
                                    c: COURSE
feature -- attribures
                                    rs: STUDENT
 students: LINKED_LIST[STUDENT]
feature -- command
                                    nrs: STUDENT
 add student(s: STUDENT)
                                    sms: SMS
   do
                                    create c.make("3311")
     students.extend(s)
                                    create sms.make
   end
 register_all (c: COURSE)
  do
    across students is s
      loop
       s.register(c)
                                    sms.add_student(rs)
     end
                                    sms.add_student(nrs)
  end
end
                                    sms.register_all(c)
```

Q: What if more kinds of students are to be introduced?







With Inheritance (Design 3) Collection of Students

```
class
 STUDENT_MANAGEMENT SYSTEM
feature -- attribures
 students: LINKED_LIST[STUDENT]
feature -- command
 add student(s: STUDENT)
   do
     students.extend(s)
   end
 register_all (c: COURSE)
   do
    across students is s
      loop
        s.register(c)
      end
   end
end
```

c: COURSE rs: STUDENT nrs: STUDENT sms: SMS create c.make("3311") create sms.make

sms.add_student(rs) sms.add_student(nrs) sms.register_all(c)

Q: What if more kinds of students are to be introduced?