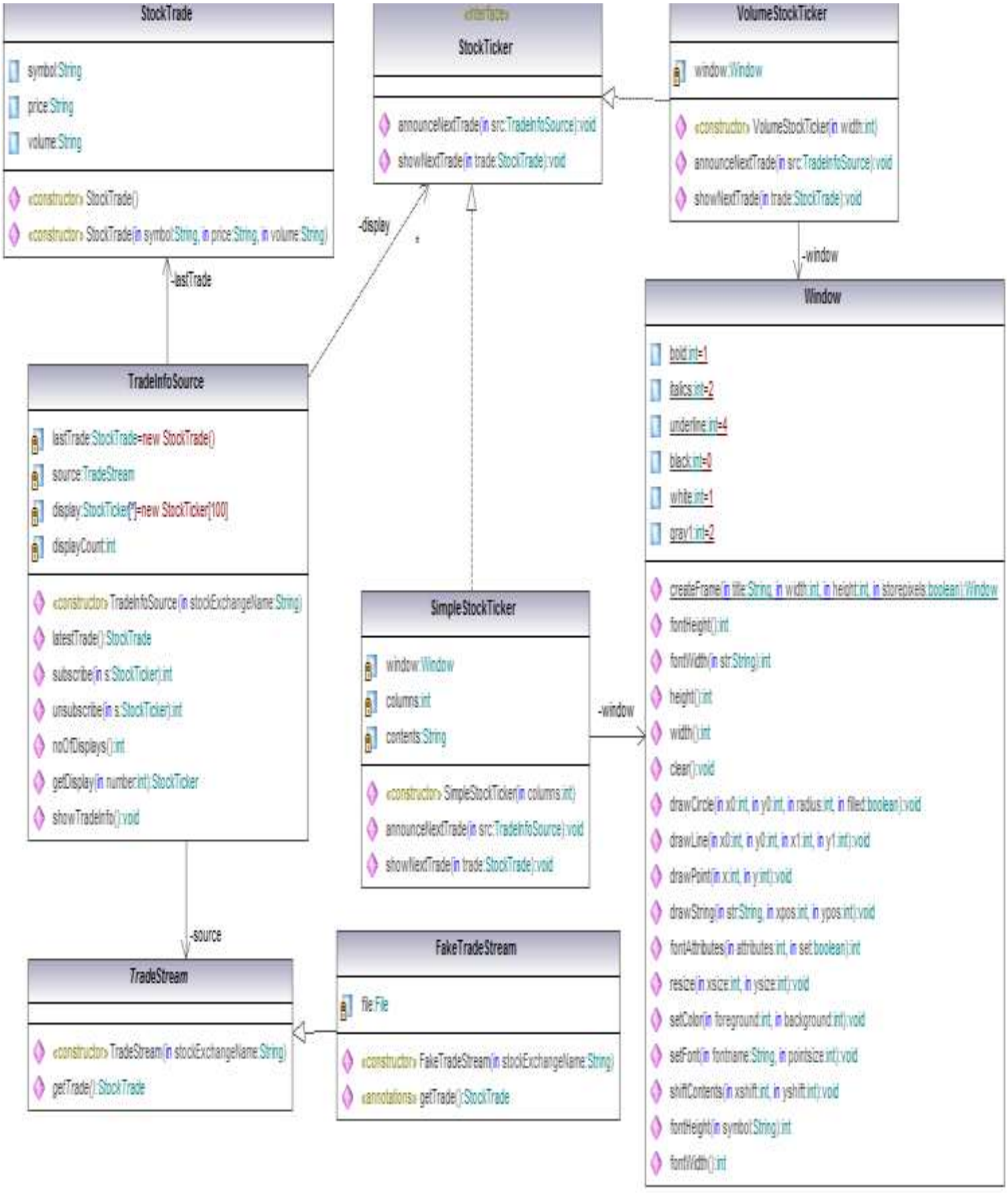
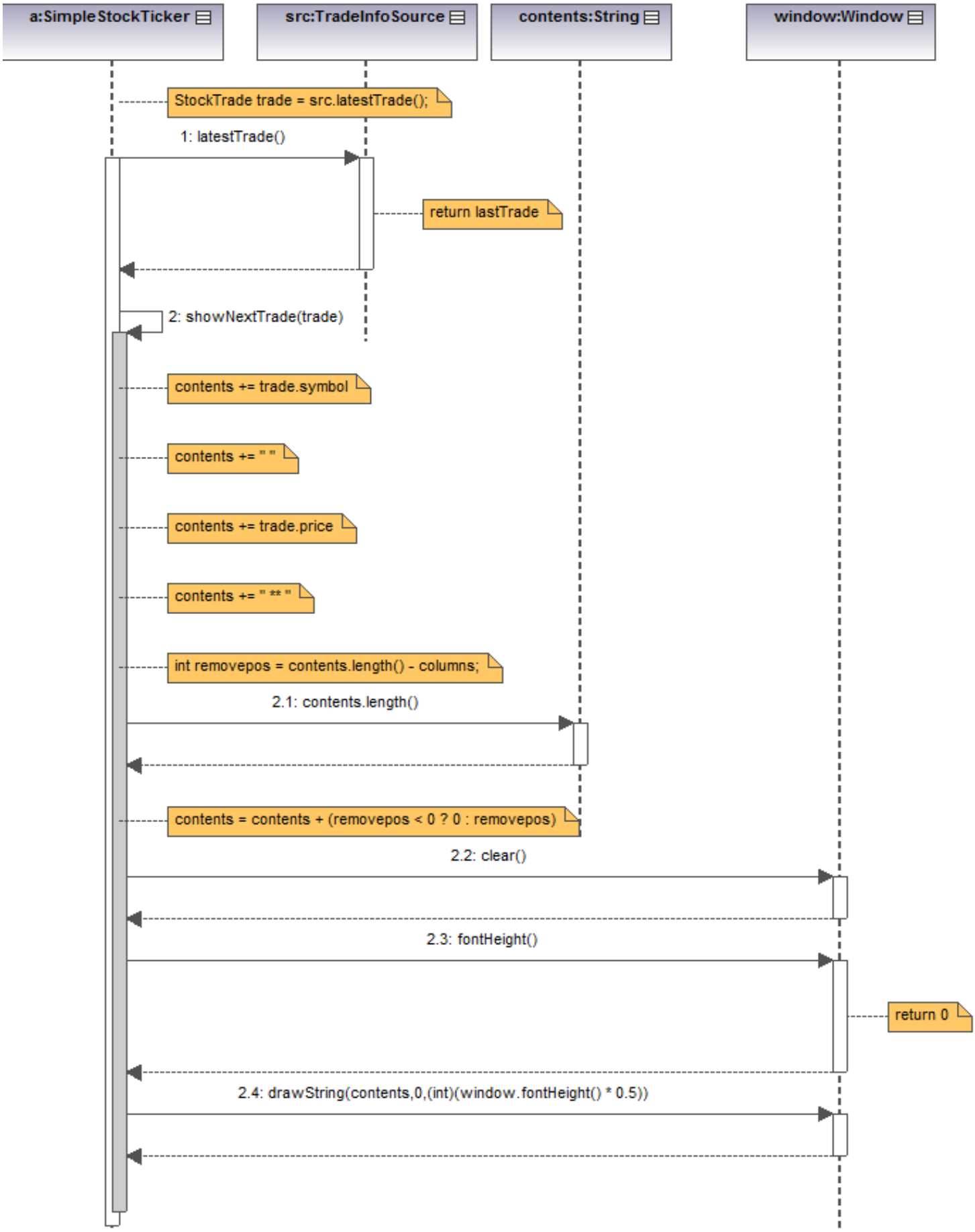
Questions: Non-Design Pattern Version

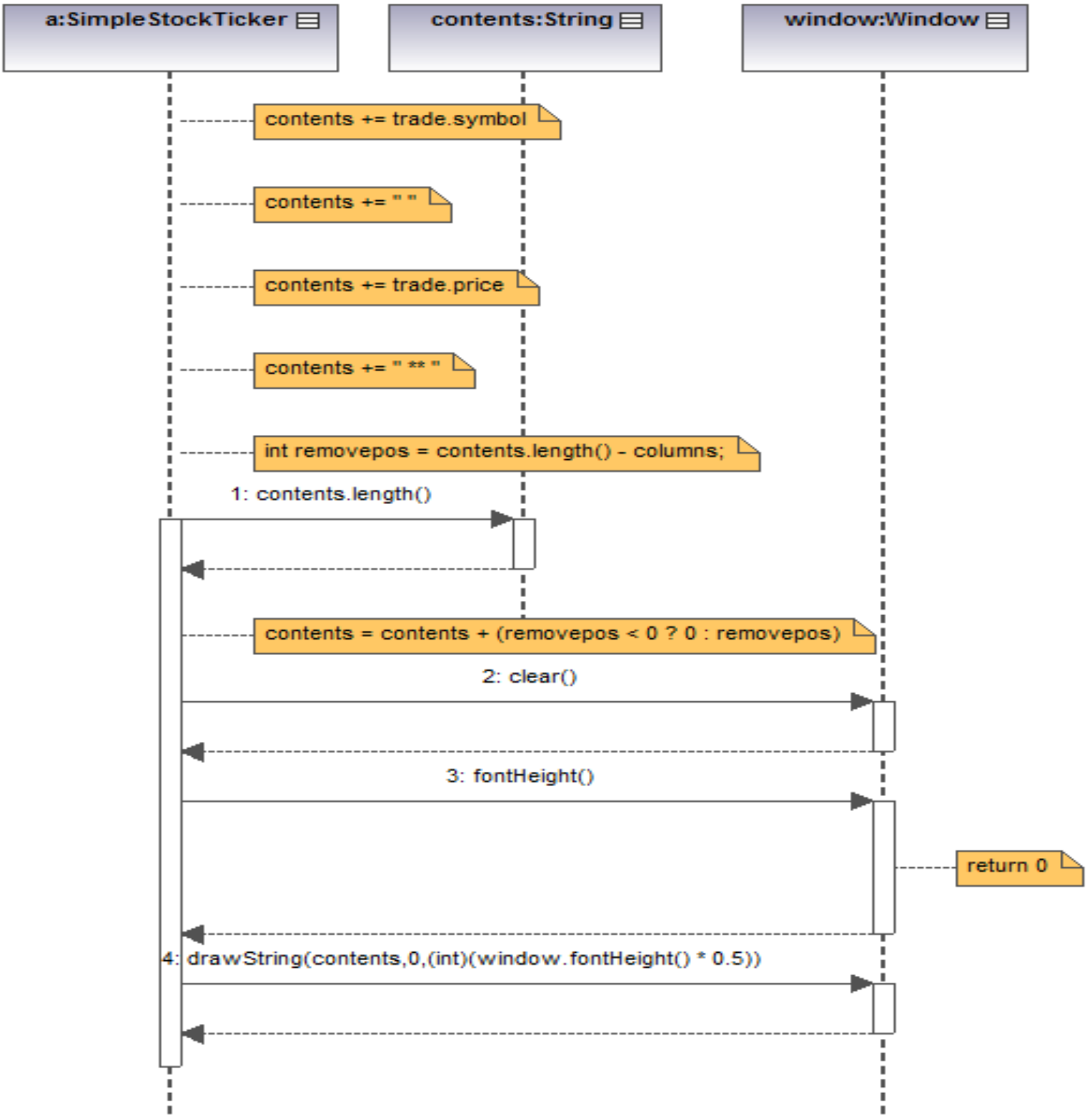
In order to complete each task, you need to fill **NAME , START TIME and END TIME** for each task.

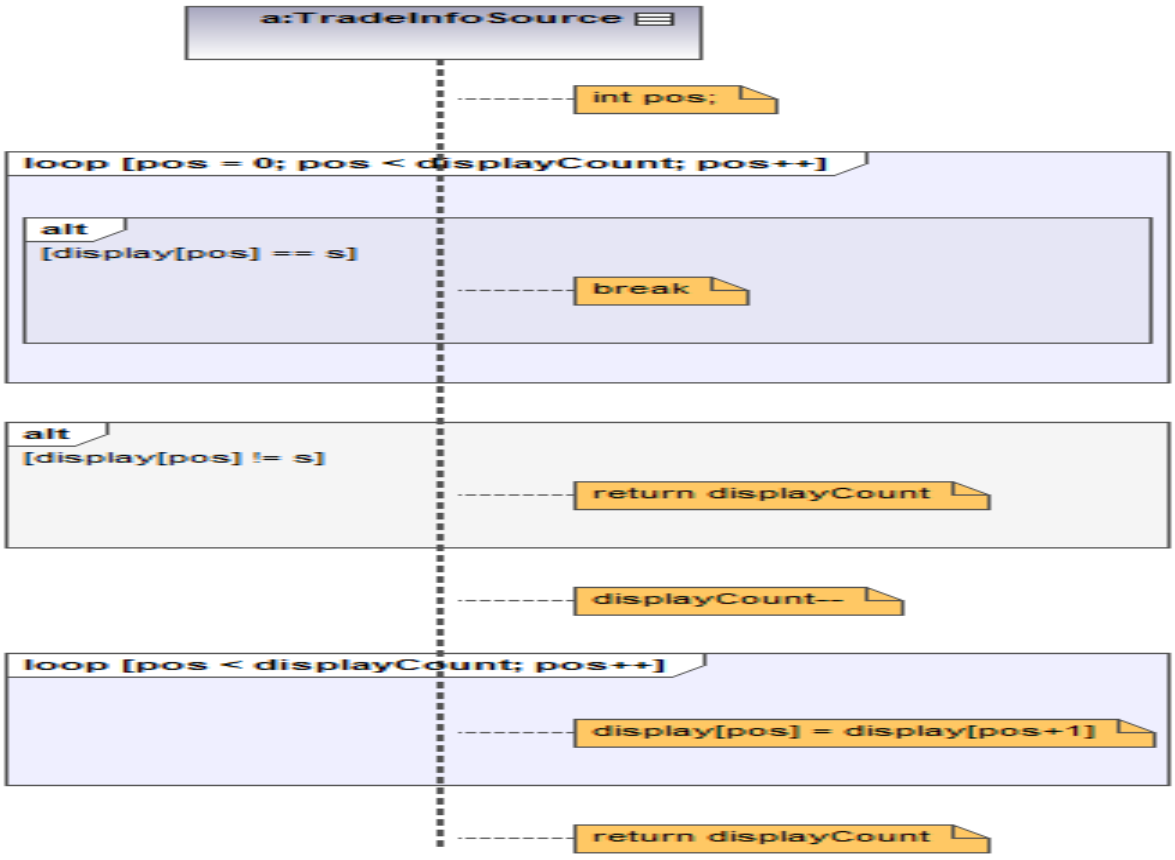
Program: Stock Ticker (Non-Design Pattern)

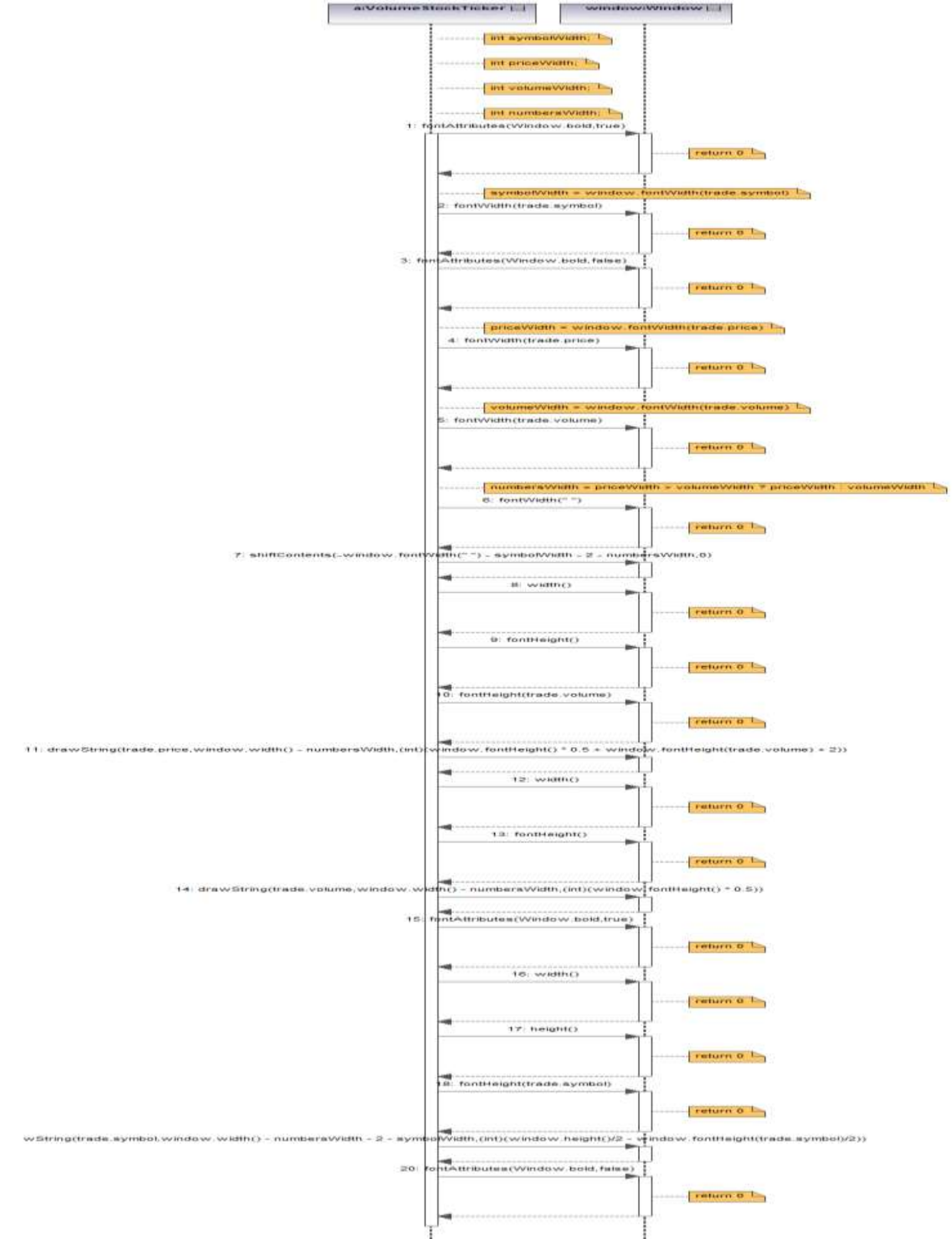
Description:
Shortly, you will receive a "stock ticker" design. This incomplete program is used for directing a continuous stream of stock trades (title, number, unit, price) from a stock market to one or more displays, which are also a part of the program. The displays advertise the information or part of it.
The data (of type *StockTrade*) come from a *FakeTradeStream* which simulates the trade datastream by reading the data repeatedly from a file. This is done for simplification.
In the present program version only *one* display shows the stock data. The type of this display is *SimpleStockTicker*. A second display named VolumeStockTicker is implemented but presently not used. Both types of display are shown in the figure below


StockTrade
symbol:String
price:String
volume:String
«constructor» StockTrade()
«constructor» StockTrade(in symbol:String, in price:String, in volume:String)
«interface»
StockTicker
announceNextTrade(in src:TradeInfoSource):void
showNextTrade(in trade:StockTrade):void
VolumeStockTicker
window:Window
«constructor» VolumeStockTicker(in width:int)
announceNextTrade(in src:TradeInfoSource):void
showNextTrade(in trade:StockTrade):void
-display
*
-lastTrade
-window
Window
bold:int=1
italics:int=2
underline:int=4
black:int=0
white:int=1
gray1:int=2
createFrame(in title:String, in width:int, in height:int, in storepixels:boolean):Window
fontHeight():int
fontWidth(in str:String):int
height():int
width():int
clear():void
drawCircle(in x0:int, in y0:int, in radius:int, in filled:boolean):void
drawLine(in x0:int, in y0:int, in x1:int, in y1:int):void
drawPoint(in x:int, in y:int):void
drawString(in str:String, in xpos:int, in ypos:int):void
font4Attributes(in attributes:int, in set:boolean):int
resize(in xsize:int, in ysize:int):void
setColor(in foreground:int, in background:int):void
setFont(in fontname:String, in pointsize:int):void
shiftContents(in xshift:int, in yshift:int):void
fontHeight(in symbol:String):int
fontWidth():int
TradeInfoSource
lastTrade:StockTrade=new StockTrade()
source:TradeStream
display:StockTicker[*]=new StockTicker[100]
displayCount:int
«constructor» TradeInfoSource(in stockExchangeName:String)
latestTrade():StockTrade
subscribe(in s:StockTicker):int
unsubscribe(in s:StockTicker):int
noOfDisplays():int
getDisplay(in number:int):StockTicker
showTradeInfo():void
SimpleStockTicker
window:Window
columns:int
contents:String
«constructor» SimpleStockTicker(in columns:int)
announceNextTrade(in src:TradeInfoSource):void
showNextTrade(in trade:StockTrade):void
-window
-source
TradeStream
«constructor» TradeStream(in stockExchangeName:String)
getTrade():StockTrade
FakeTradeStream
file:File
«constructor» FakeTradeStream(in stockExchangeName:String)
«annotations» getTrade():StockTrade


a:SimpleStockTicker
src:TradeInfoSource
contents:String
window:Window
StockTrade trade = src.latestTrade();
1: latestTrade()
return lastTrade
2: showNextTrade(trade)
contents += trade.symbol
contents += " "
contents += trade.price
contents += " ** "
int removepos = contents.length() - columns;
2.1: contents.length()
contents = contents + (removepos < 0 ? 0 : removepos)
2.2: clear()
2.3: fontHeight()
return 0
2.4: drawString(contents,0,(int)(window.fontHeight() * 0.5))


a:SimpleStockTicker
contents:String
window:Window
contents += trade.symbol
contents += " "
contents += trade.price
contents += " ** "
int removepos = contents.length() - columns;
1: contents.length()
contents = contents + (removepos < 0 ? 0 : removepos)
2: clear()
3: fontHeight()
return 0
4: drawString(contents,0,(int)(window.fontHeight() * 0.5))


a:TradeInfoSource
int pos;
loop [pos = 0; pos < displayCount; pos++]
alt
[display[pos] == s]
break
alt
[display[pos] != s]
return displayCount
displayCount--
loop [pos < displayCount; pos++]
display[pos] = display[pos+1]
return displayCount


a:VolumeStockTicker
window:Window
int symbolWidth;
int priceWidth;
int volumeWidth;
int numbersWidth;
1: fontAttributes(Window.bold,true)
return 0
symbolWidth = window.fontWidth(trade.symbol)
2: fontWidth(trade.symbol)
return 0
3: fontAttributes(Window.bold,false)
return 0
priceWidth = window.fontWidth(trade.price)
4: fontWidth(trade.price)
return 0
volumeWidth = window.fontWidth(trade.volume)
5: fontWidth(trade.volume)
return 0
numbersWidth = priceWidth + volumeWidth + priceWidth + volumeWidth
6: fontWidth(" ")
return 0
7: shiftContents(-window.fontWidth(" ") - symbolWidth - 2 - numbersWidth,0)
8: width()
return 0
9: fontHeight()
return 0
10: fontHeight(trade.volume)
return 0
11: drawString(trade.price,window.width() - numbersWidth,(int)(window.fontHeight() * 0.5 + window.fontHeight(trade.volume) + 2))
12: width()
return 0
13: fontHeight()
return 0
14: drawString(trade.volume,window.width() - numbersWidth,(int)(window.fontHeight() * 0.5))
15: fontAttributes(Window.bold,true)
return 0
16: width()
return 0
17: height()
return 0
18: fontHeight(trade.symbol)
return 0
wString(trade.symbol,window.width() - numbersWidth - 2 - symbolWidth,(int)(window.height()/2 - window.fontHeight(trade.symbol)/2))
20: fontAttributes(Window.bold,false)
return 0


arVolumeStockTicker [:]
src:TradeInfoSource [:]
Window:Window [:]
StockTrade trade = src.latestTrade();
1: latestTrade()
return lastTrade ;
2: showNextTrade(trade)
int symbolWidth;
int priceWidth;
int volumeWidth;
int numbersWidth;
2.1: fontAttributes(Window.bold,true)
return 0 ;
symbolWidth = window.fontWidth(trade.symbol) ;
2.2: fontWidth(trade.symbol)
return 0 ;
2.3: fontAttributes(Window.bold,false)
return 0 ;
priceWidth = window.fontWidth(trade.price) ;
2.4: fontWidth(trade.price)
return 0 ;
volumeWidth = window.fontWidth(trade.volume) ;
2.5: fontWidth(trade.volume)
return 0 ;
numbersWidth = priceWidth + volumeWidth * priceWidth + volumeWidth ;
2.6: fontWidth(" ")
return 0 ;
2.7: shiftContents(-window.fontWidth(" ") - symbolWidth - 2 - numbersWidth,0)
2.8: width()
return 0 ;
2.9: fontHeight()
return 0 ;
2.10: fontHeight(trade.volume)
return 0 ;
2.11: drawString(trade.price,window.width() - numbersWidth,(int)(window.fontHeight() * 0.5 + window.fontHeight(trade.volume) + 2))
2.12: width()
return 0 ;
2.13: fontHeight()
return 0 ;
2.14: drawString(trade.volume,window.width() - numbersWidth,(int)(window.fontHeight() * 0.5))
2.15: fontAttributes(Window.bold,true)
return 0 ;
2.16: width()
return 0 ;
2.17: height()
return 0 ;
2.18: fontHeight(trade.symbol)
return 0 ;
drawString(trade.symbol,window.width() - numbersWidth - 2 - symbolWidth,(int)(window.height()/2 - window.fontHeight(trade.symbol)/2))
2.20: fontAttributes(Window.bold,false)
return 0 ;

Task 1: Please answer all these questions.

Name: _____

Start Time: ____:____ **(hh : mm)** ** The time that you begin answer the question.

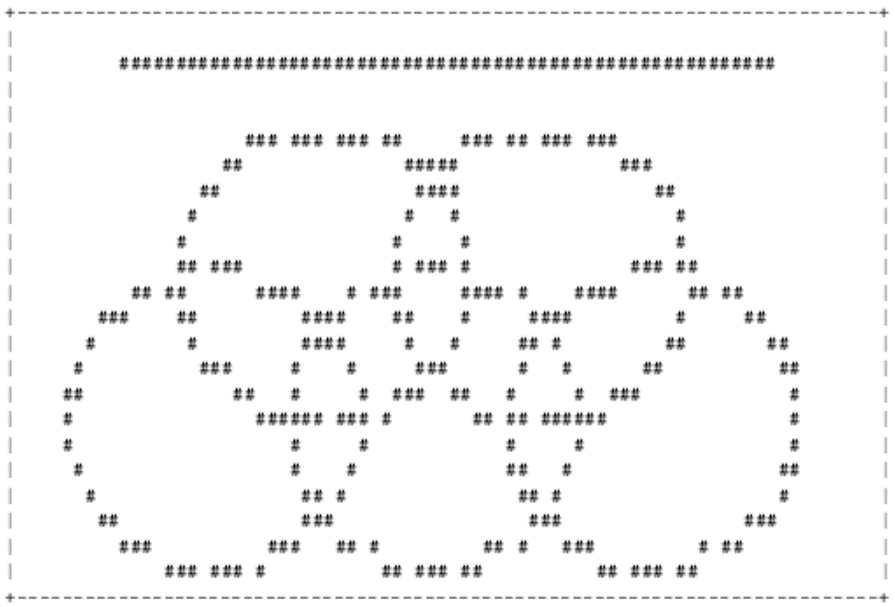
1. Provide all class names that access the TradeInfoSource class for getting the data.
   _____
2. Provide all class names that access the access the createFrame method in the Window class.
   _____
3. Provide the class names that are invoked by a constructor of the TradeInfoSource class.
   _____

End Time: ____:____ **(hh : mm)** ** The time that you completely answer all questions.

Program: Graphics Library (Non-Design Pattern)

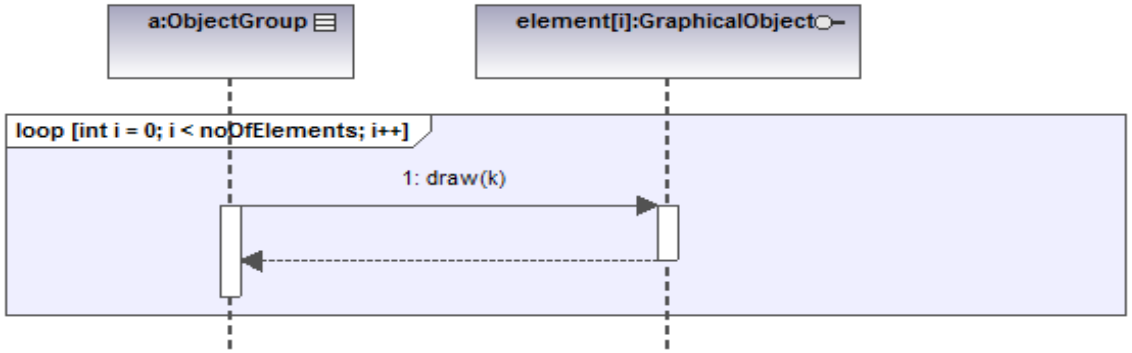
Description

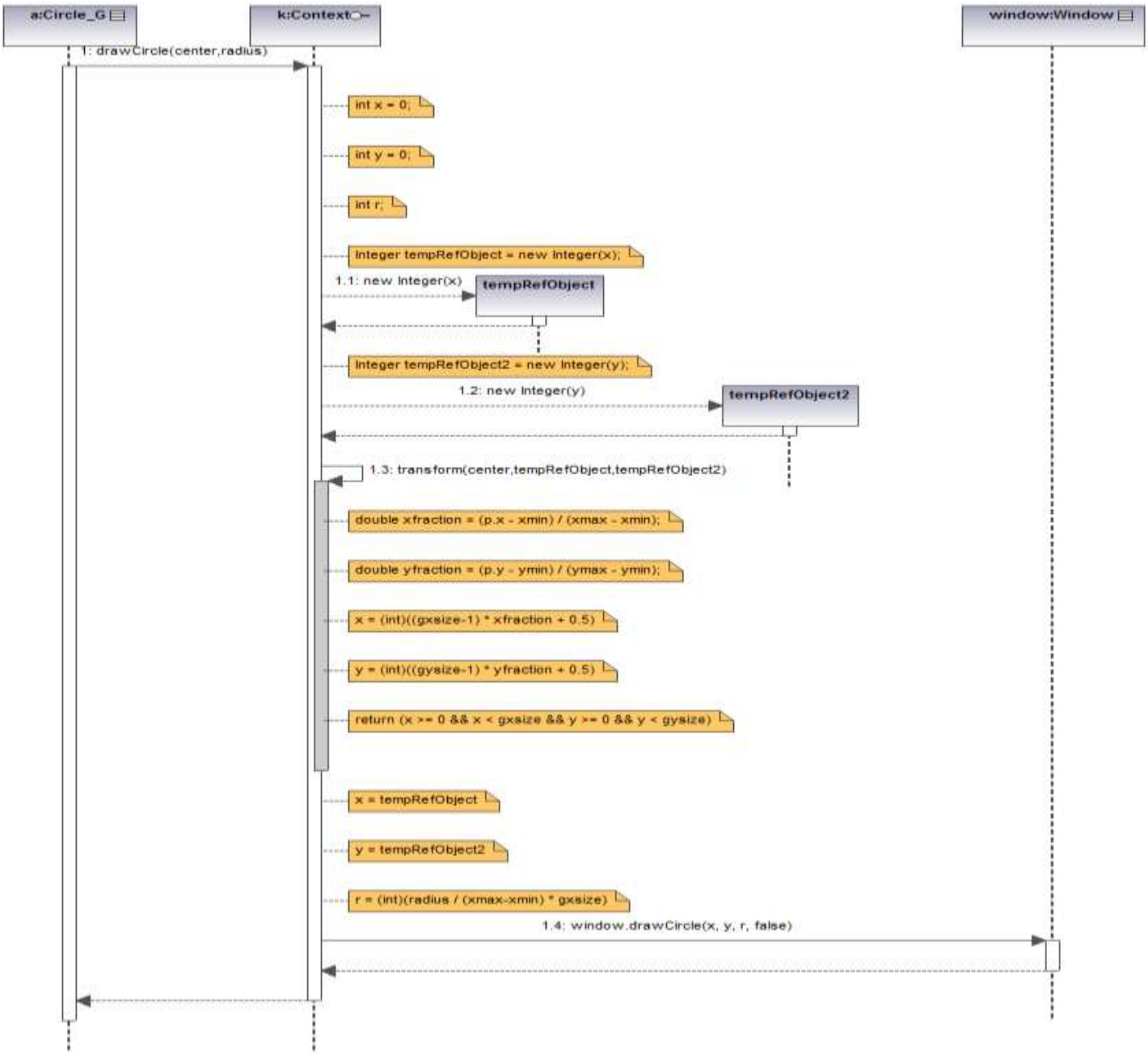
You will receive a "graphics library" design. It contains a library for creating, manipulating, and drawing simple types of graphical objects (*Line, Circle*) on different types of graphical devices (alpha-numerical output *A*, pixel graphic device *G*). Furthermore it contains a simple main program that uses some of the objects and operations. It displays the result shown below.

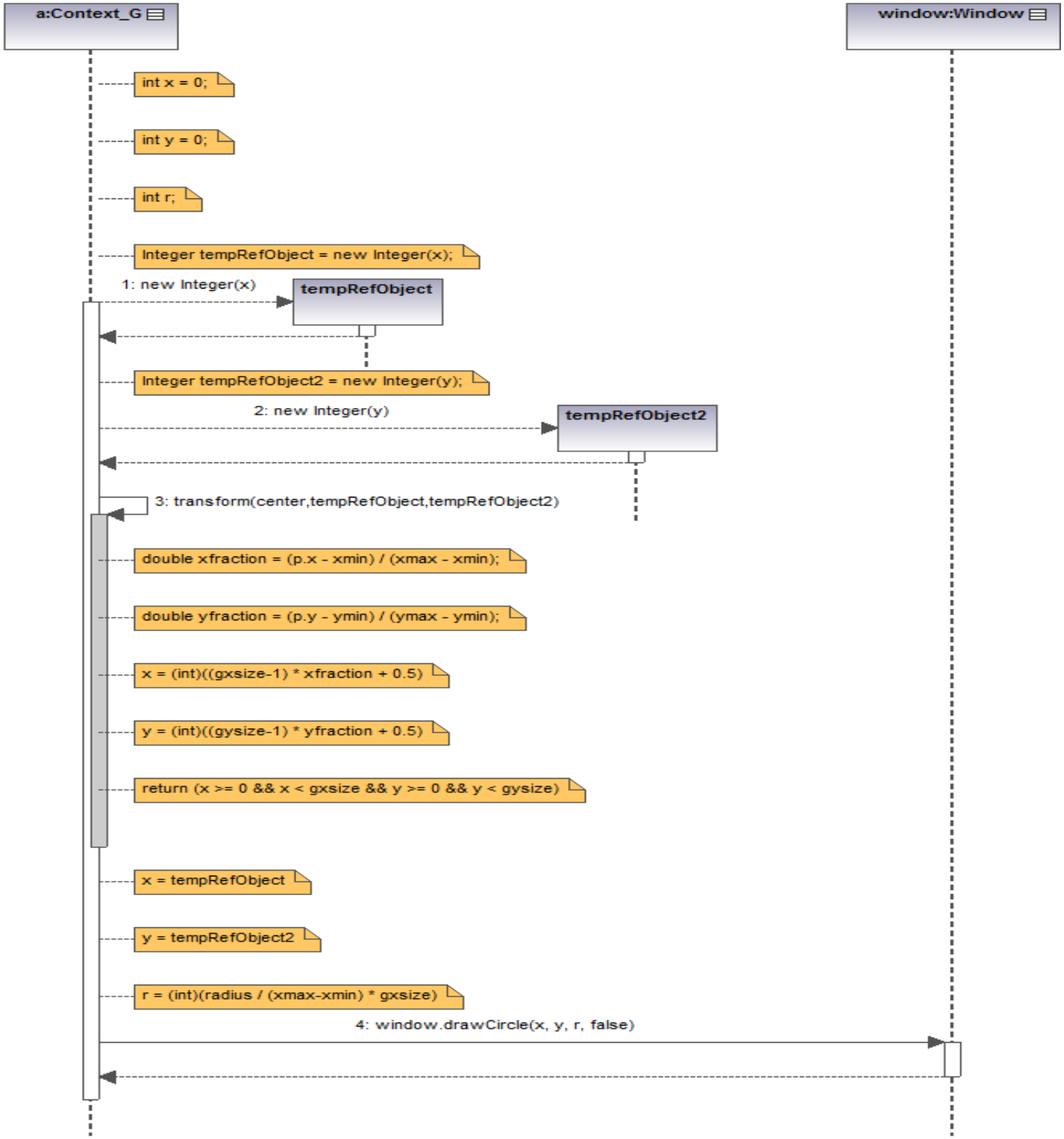


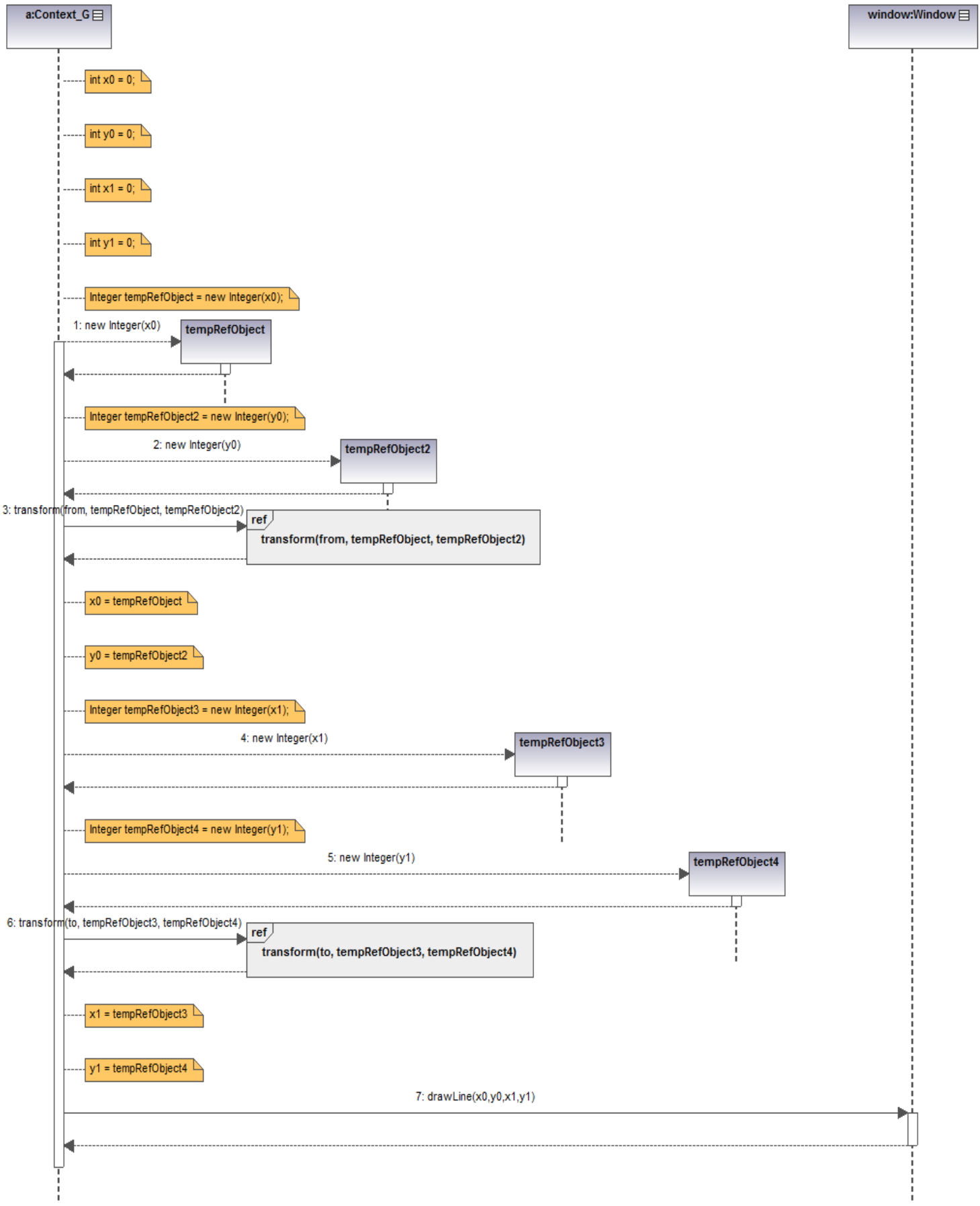
A collection of classes (*Context*) implements the primitive operations for different output devices. There is a minimal set of operations but some devices provide additional functionality. For this reason some objects (in this case *Line*) are implemented only once for all devices but other objects (in this case *Circle*) are implemented individually for each device. Which device is to be used can be selected at a central spot *Generator*.

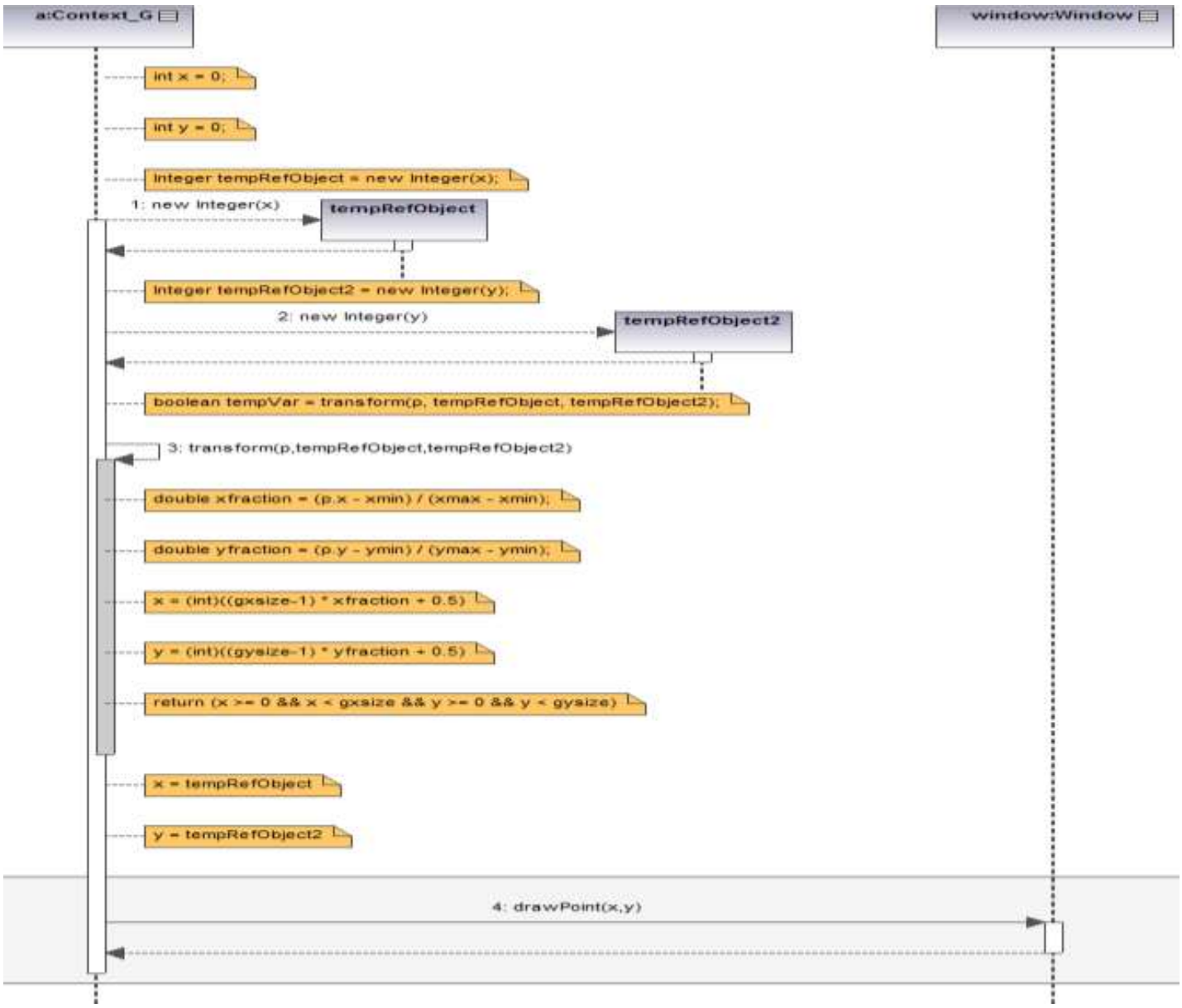

Context_A
gxsize:int
gysize:int
drawingarea:char[][]
xmin:double
xmax:double
ymin:double
ymax:double
«constructor» Context_A()
«constructor» Context_A(in gxsize:int, in gysize:int, in xmin:double, in ymin:double, in xmax:double, in ymax:double)
xResolution():double
yResolution():double
erase():void
drawPoint(in p:Point):void
drawLine(in fromPoint, in to:Point):void
display():void
Generator
type:Generatortype
newGenerator(in e:Generatortype):Generator
«constructor» Generator(in t:Generatortype)
newContext():Context
newContext(in gxsize:int, in gysize:int, in xmin:double, in ymin:double, in xmax:double, in ymax:double):Context
newObjectgroup():ObjectGroup
newLine(in fromPoint, in to:Point):Line
newCircle(in centerPoint, in radius:double):Circle
setType(in type:Generatortype):void
getType():Generatortype
-type
«enumeration»
Generatortype
«interface»
Context
erase():void
xResolution():double
yResolution():double
drawPoint(in NamelessParameter:Point):void
drawLine(in NamelessParameter1:Point, in NamelessParameter2:Point):void
display():void
Window
bold:int=1
italics:int=2
underline:int=4
black:int=0
white:int=1
gray1:int=2
createFrame(in title:String, in width:int, in height:int, in storeopens:boolean):Window
fontHeight(in str:String):int
fontWidth(in str:String):int
height():int
width():int
clear():void
drawCircle(in x0:int, in y0:int, in radius:int, in filled:boolean):void
drawLine(in x0:int, in y0:int, in x1:int, in y1:int):void
drawPoint(in x:int, in y:int):void
drawString(in str:char, in xpos:int, in ypos:int):void
fontAttributes(in attributes:int, in set:boolean):int
resize(in xsize:int, in ysize:int):void
setColor(in foreground:int, in background:int):void
setFont(in fontname:char, in pointsize:int):void
shiftContents(in xshift:int, in yshift:int):void
Context_G
gxsize:int
gysize:int
xmin:double
xmax:double
ymin:double
ymax:double
window:Window
transform(in p:Point, in x:Integer, in y:Integer):boolean
«constructor» Context_G()
«constructor» Context_G(in gxsize:int, in gysize:int, in xmin:double, in ymin:double, in xmax:double, in ymax:double)
xResolution():double
yResolution():double
erase():void
drawPoint(in p:Point):void
drawLine(in fromPoint, in to:Point):void
display():void
drawCircle(in centerPoint, in radius:double):void
-window
textrun
main(in args:String[*]):void
cross(in generator:Generator):void
«interface»
GraphicalObject
rotate(in centerPoint, in angle:double):void
shift(in dx:double, in dy:double):void
draw(in k:Context):void
Line
startPoint=new Point()
endPoint=new Point()
«constructor» Line()
«constructor» Line(in startPoint, in endPoint)
rotate(in centerPoint, in angle:double):void
shift(in dx:double, in dy:double):void
draw(in k:Context):void
-element
*
ObjectGroup
limit:int=100
element:GraphicalObject[*]=new GraphicalObject[limit]
noOfElements:int
«constructor» ObjectGroup()
add(in k:GraphicalObject):void
rotate(in centerPoint, in angle:double):void
shift(in dx:double, in dy:double):void
draw(in k:Context):void
Circle
centerPoint=new Point()
radius:double
«constructor» Circle()
«constructor» Circle(in centerPoint, in radius:double)
rotate(in centerPoint, in angle:double):void
shift(in dx:double, in dy:double):void
draw(in k:Context):void
-start
-end
#center
Point
x:double
y:double
«constructor» Point()
«constructor» Point(in x:double, in y:double)
rotate(in centerPoint, in angle:double):void
shift(in dx:double, in dy:double):void
Circle_A
«constructor» Circle_A()
«constructor» Circle_A(in centerPoint, in radius:double)
draw(in k:Context):void
Circle_G
«constructor» Circle_G()
«constructor» Circle_G(in centerPoint, in radius:double)
draw(in k:Context):void

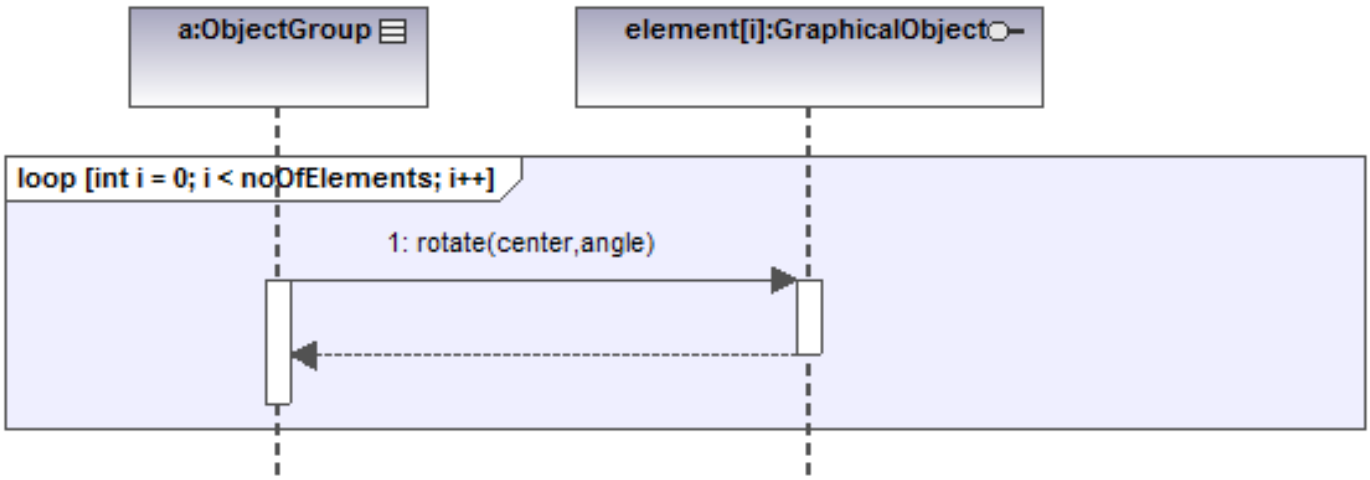

a:ObjectGroup
element[i]:GraphicalObject
loop [int i = 0; i < noOfElements; i++]
1: draw(k)

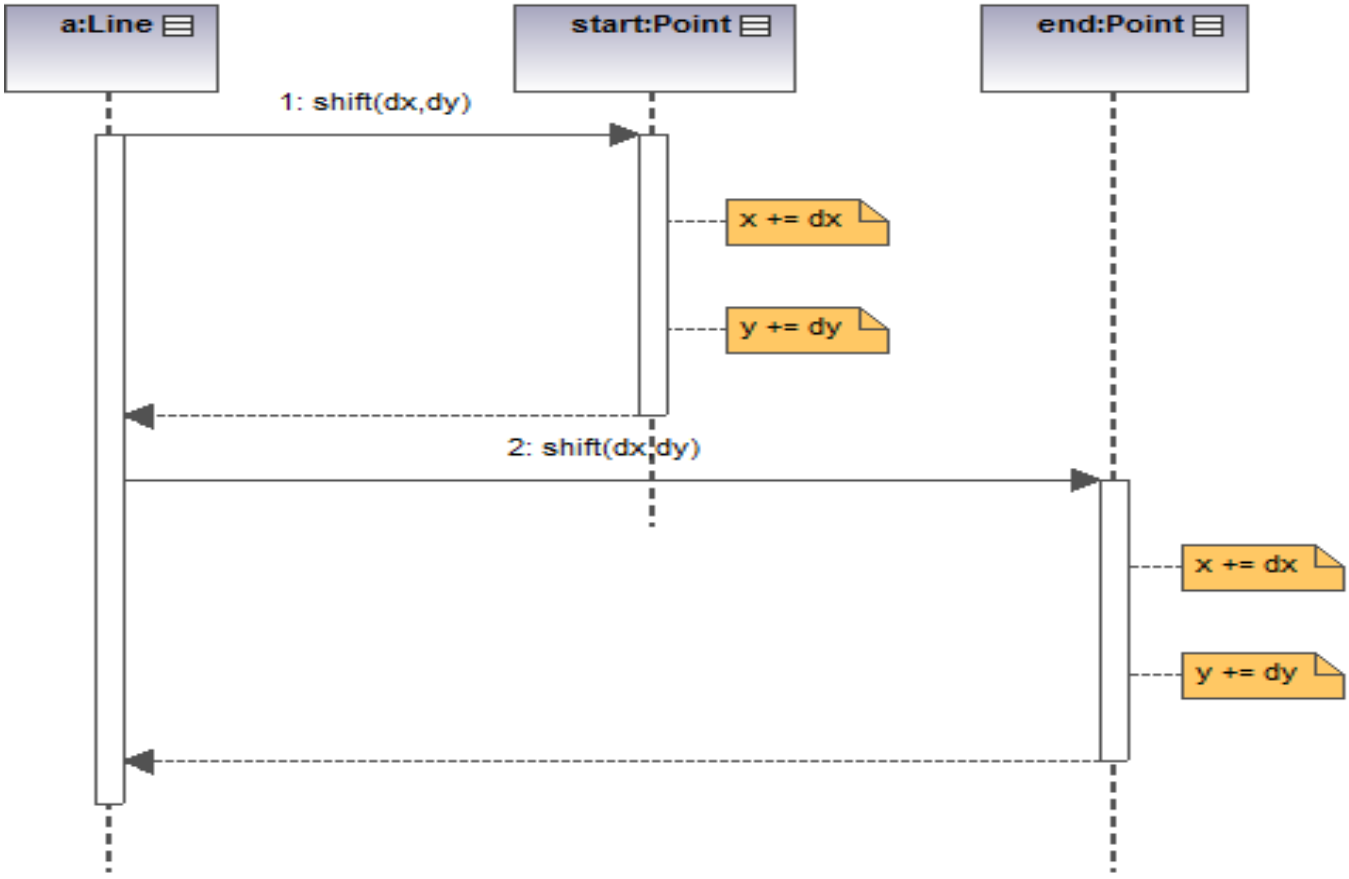

a:Circle_G
k:Context
window:Window
1: drawCircle(center,radius)
int x = 0;
int y = 0;
int r;
Integer tempRefObject = new Integer(x);
1.1: new Integer(x)
tempRefObject
Integer tempRefObject2 = new Integer(y);
1.2: new Integer(y)
tempRefObject2
1.3: transform(center,tempRefObject,tempRefObject2)
double xfraction = (p.x - xmin) / (xmax - xmin);
double yfraction = (p.y - ymin) / (ymax - ymin);
x = (int)((gxsize-1) * xfraction + 0.5)
y = (int)((gysize-1) * yfraction + 0.5)
return (x >= 0 && x < gxsize && y >= 0 && y < gysize)
x = tempRefObject
y = tempRefObject2
r = (int)(radius / (xmax-xmin) * gxsize)
1.4: window.drawCircle(x, y, r, false)


a:Context_G
window:Window
int x = 0;
int y = 0;
int r;
Integer tempRefObject = new Integer(x);
1: new Integer(x)
tempRefObject
Integer tempRefObject2 = new Integer(y);
2: new Integer(y)
tempRefObject2
3: transform(center,tempRefObject,tempRefObject2)
double xfraction = (p.x - xmin) / (xmax - xmin);
double yfraction = (p.y - ymin) / (ymax - ymin);
x = (int)((gxsize-1) * xfraction + 0.5)
y = (int)((gysize-1) * yfraction + 0.5)
return (x >= 0 && x < gxsize && y >= 0 && y < gysize)
x = tempRefObject
y = tempRefObject2
r = (int)(radius / (xmax-xmin) * gxsize)
4: window.drawCircle(x, y, r, false)


a:Context_G
window:Window
int x0 = 0;
int y0 = 0;
int x1 = 0;
int y1 = 0;
Integer tempRefObject = new Integer(x0);
1: new Integer(x0)
tempRefObject
Integer tempRefObject2 = new Integer(y0);
2: new Integer(y0)
tempRefObject2
3: transform(from, tempRefObject, tempRefObject2)
ref
transform(from, tempRefObject, tempRefObject2)
x0 = tempRefObject
y0 = tempRefObject2
Integer tempRefObject3 = new Integer(x1);
4: new Integer(x1)
tempRefObject3
Integer tempRefObject4 = new Integer(y1);
5: new Integer(y1)
tempRefObject4
6: transform(to, tempRefObject3, tempRefObject4)
ref
transform(to, tempRefObject3, tempRefObject4)
x1 = tempRefObject3
y1 = tempRefObject4
7: drawLine(x0,y0,x1,y1)


a:Context_G
window:Window
int x = 0;
int y = 0;
Integer tempRefObject = new Integer(x);
1: new Integer(x)
tempRefObject
Integer tempRefObject2 = new Integer(y);
2: new Integer(y)
tempRefObject2
boolean tempVar = transform(p, tempRefObject, tempRefObject2);
3: transform(p,tempRefObject,tempRefObject2)
double xfraction = (p.x - xmin) / (xmax - xmin);
double yfraction = (p.y - ymin) / (ymax - ymin);
x = (int)((gxsize-1) * xfraction + 0.5);
y = (int)((gysize-1) * yfraction + 0.5);
return (x >= 0 && x < gxsize && y >= 0 && y < gysize)
x = tempRefObject
y = tempRefObject2
4: drawPoint(x,y)


a:ObjectGroup
element[i]:GraphicalObject
loop [int i = 0; i < noOfElements; i++]
1: rotate(center,angle)


a:Line
start:Point
end:Point
1: shift(dx,dy)
x += dx
y += dy
2: shift(dx,dy)
x += dx
y += dy

Task 2: Please answer all these questions.

Name: _____

Start Time: ____:____ **(hh : mm)** ** The time that you begin answer the question.

1. Provide all class names that plot the 2D point.
   _____
2. Provide the class name that control types of graphical.
   _____
3. Provide all class names that manipulate the GraphicalObject objects.
   _____
4. From the Question 3, provide all method names that collect the GraphicalObject objects.
   _____
5. Provide a base class of the graphical representation hierarchy.
   _____
6. Provide all class names that the Circle_A class uses to draw a circle.
   _____
7. Provide all class names that the Line class uses to draw a line.
   _____
8. Provide all class names that display circle on device G.
   _____
9. Provide all class names that call the drawCircle method in the Window class.
   _____
10. Provide all class names that invoke the Generator class.
    _____

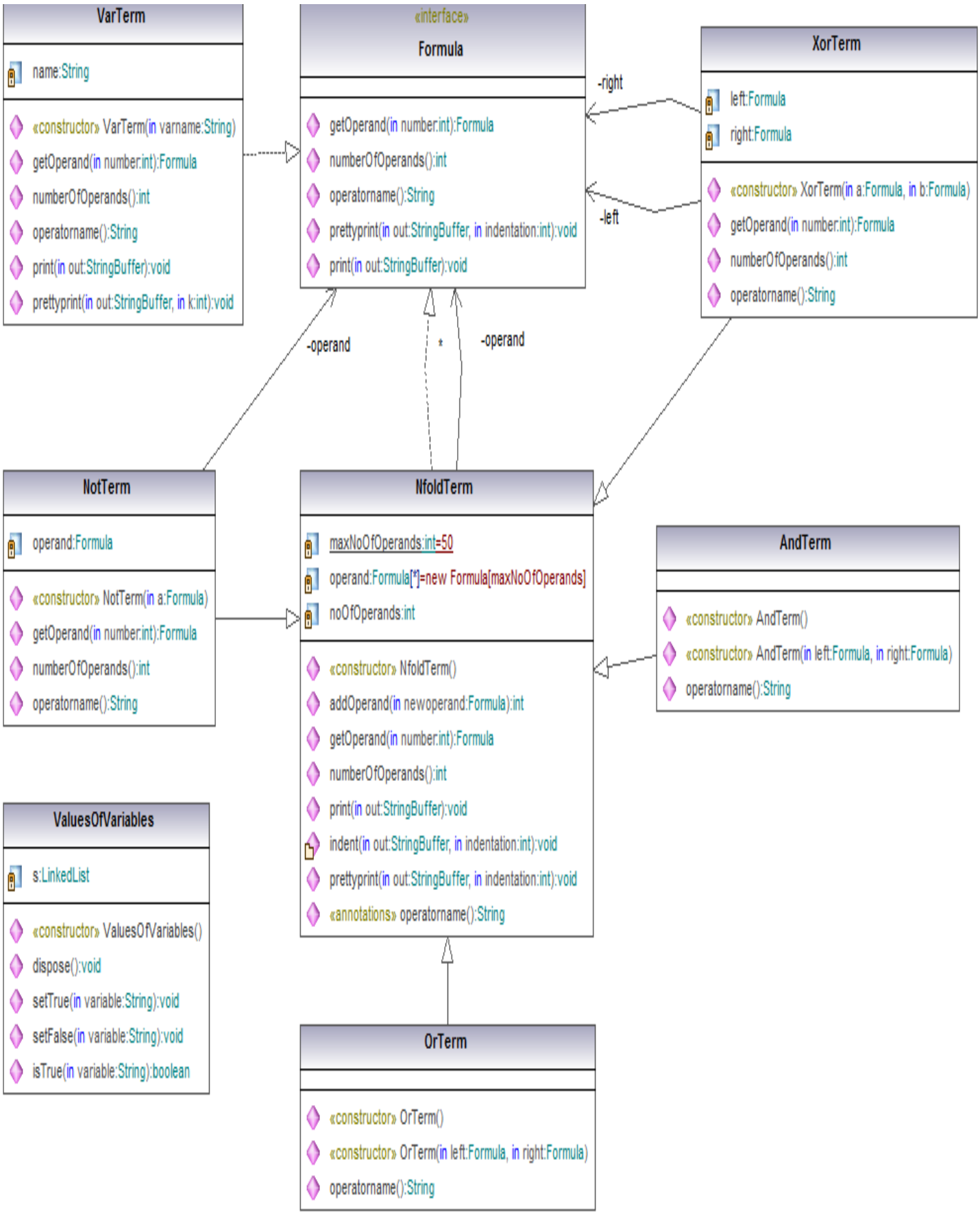
End Time: ____:____ **(hh : mm)** ** The time that you completely answer all questions.

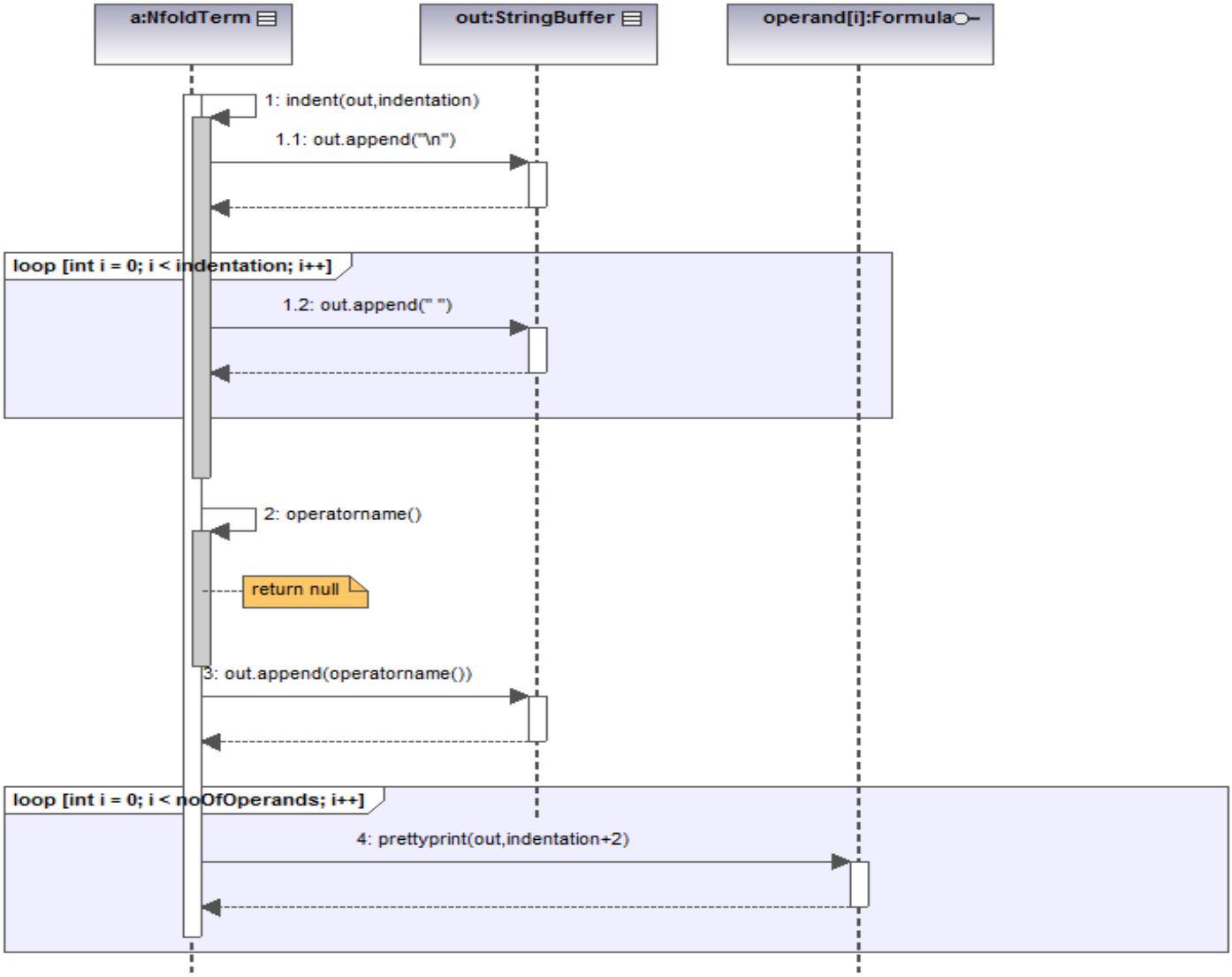
Program: Boolean Formulas

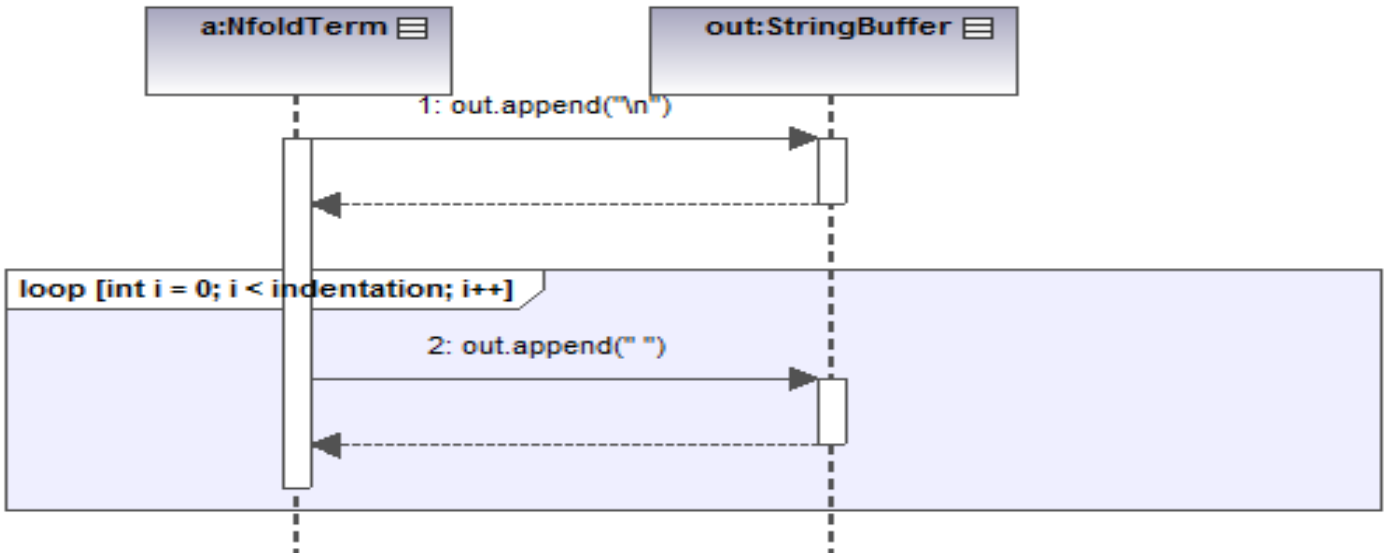
Description:
You will receive the "boolean formulas" design. It contains a library for representing boolean formulas (using AND, OR, XOR, NOT, and variables) and for printing the formulas in two different styles: in infix notation on a single line or prefix notation on multiple lines with indentations. Furthermore, it contains a small main program, which generates a formula and invokes both printing routines this leads to the following

```
((a XOR NOT b) AND (NOT x1 OR NOT x2))

AND
  XOR
    a
    NOT
      b
  OR
    NOT
      x1
    NOT
      x2
```


VarTerm
name:String
«constructor» VarTerm(in varname:String)
getOperand(in number:int):Formula
numberOfOperands():int
operatorname():String
print(in out:StringBuffer):void
prettyprint(in out:StringBuffer, in k:int):void
«interface»
Formula
getOperand(in number:int):Formula
numberOfOperands():int
operatorname():String
prettyprint(in out:StringBuffer, in indentation:int):void
print(in out:StringBuffer):void
-right
-left
XorTerm
left:Formula
right:Formula
«constructor» XorTerm(in a:Formula, in b:Formula)
getOperand(in number:int):Formula
numberOfOperands():int
operatorname():String
-operand
*
-operand
NotTerm
operand:Formula
«constructor» NotTerm(in a:Formula)
getOperand(in number:int):Formula
numberOfOperands():int
operatorname():String
NfoldTerm
maxNoOfOperands:int=50
operand:Formula[*]=new Formula[maxNoOfOperands]
noOfOperands:int
«constructor» NfoldTerm()
addOperand(in newoperand:Formula):int
getOperand(in number:int):Formula
numberOfOperands():int
print(in out:StringBuffer):void
indent(in out:StringBuffer, in indentation:int):void
prettyprint(in out:StringBuffer, in indentation:int):void
«annotations» operatorname():String
AndTerm
«constructor» AndTerm()
«constructor» AndTerm(in left:Formula, in right:Formula)
operatorname():String
ValuesOfVariables
s:LinkedList
«constructor» ValuesOfVariables()
dispose():void
setTrue(in variable:String):void
setFalse(in variable:String):void
isTrue(in variable:String):boolean
OrTerm
«constructor» OrTerm()
«constructor» OrTerm(in left:Formula, in right:Formula)
operatorname():String


a:NfoldTerm
out:StringBuffer
operand[i]:Formula
1: indent(out,indentation)
1.1: out.append("\n")
loop [int i = 0; i < indentation; i++]
1.2: out.append(" ")
2: operatorname()
return null
3: out.append(operatorname())
loop [int i = 0; i < noOfOperands; i++]
4: prettyprint(out,indentation+2)


a:NfoldTerm
out:StringBuffer
1: out.append("\n")
loop [int i = 0; i < indentation; i++]
2: out.append(" ")

Task 3: Please answer all these questions.

Name: _____

Start Time: ____:____ **(hh : mm)** ** The time that you begin answer the question.

1. List the class name(s) that access numberOfOperands function in the class AndTerm.
   _____
2. List the class function name(s) that are used in a constructor of the class OrTerm.
   _____
3. List the value that is returned from getOperand function in the class VerTerm.
   _____
4. List the value that is returned from numberOfOperands function in the class XorTerm.
   _____
5. List the value that is returned from numberOfOeprands function in the VarTerm.
   _____
6. List the value that is returned from numberOfOperands function in the class NotTerm.
   _____

End Time: ____:____ **(hh : mm)** ** The time that you completely answer all questions.

Program: Communication Channel Library

Description:

Shortly, you will receive a "communication channels" design.
It consists of a library for configuring and running a communication channel. Such a channel CommChannel establishes a communication connection and one can add logging, data compression, and encryption.
The library does not implement the functionality itself, instead it implements a simplified interface for the *combination* of the parts. The names of the operations indicate approximately the functionality; an exact definition is not necessary for the task.


CommChannel
channelState:State
partner:String
channel:Socket
doCompression:boolean
doEncryption:boolean
encrypt:EncryptionStream
decrypt:DecryptionStream
doLogging:boolean
logname:String
log:String
basicOpen():Result
«constructor» CommChannel(in partner:String, in compress:boolean, in key:String, in logchannelname:String)
open():Result
transmit(in d:Datablock):Result
receive(in d:Datablock):Result
reset():Result
close():Result
State
Result
-decrypt
-encrypt
-channel
DecryptionStream
«constructor» DecryptionStream(in key:String)
decrypt(in cryptext:Datablock, in cleartext:Datablock):void
decryptedsize(in encryptedtextsize:int):int
reset():void
EncryptionStream
«constructor» EncryptionStream(in key:String)
encrypt(in cleartext:Datablock, in cryptext:Datablock):void
encryptedsize(in cleartextsize:int):int
reset():void
Socket
open():int
close():int
read(in d:Datablock):int
write(in d:Datablock):int
clear():int
Datablock
«constructor» Datablock()
«constructor» Datablock(in expectedSize:int)
size():int
fingerprint():String
Compressor
compress(in plain:Datablock, in compressed:Datablock):int
decompress(in compressed:Datablock, in plain:Datablock):int
CommServices
getPartner(in partnerDescriptor:String):Socket
Logging

Task 4: Please answer all these questions.

Name: _____

Start Time: ____:____ **(hh : mm)** ** The time that you begin answer the question.

1. Provide all class name that have a function to decrypt the data.
_____

2. Provide all class names that have a function to encrypt the data.
_____

3. Provide all class names that have a function to compress the data.
_____

4. Provide all class names that represent the unit of the data between two computers.
_____

5. Provide all class names that control the communication channel.
_____

6. Provide all class names that use the function in the Logging class.
_____

7. Provide all class names that provide the stream object for writing a log file.
_____

8 In the CommChannel class, please provide all method names that are invoked by the Datablock class.
_____

9. Provide all class names that do not invoke the Dtablock class.
_____

10. Provide all class names that create a Sockectfor making a connection.
_____

End Time: ____:____ **(hh : mm)** ** The time that you completely answer all questions.