

Assignment #1

Classes

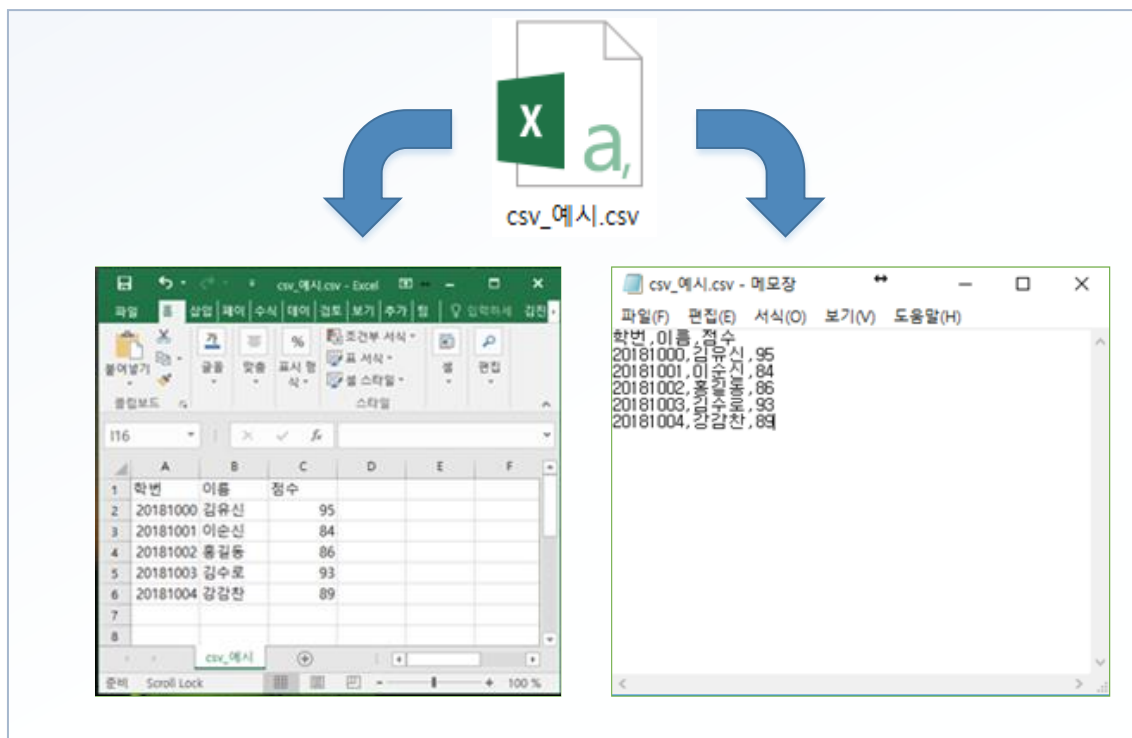
Due date: ~4/3(Tue.) 23:59

1 본 과제에서는 "CSV Reader와 SVG 변환 프로그램"을 구현한다. 이를 통하여 클래스를 이용한 객체 지향 프로그래밍의 기초를 연습한다.

1. 배경지식

1) CSV 파일 형식

CSV (comma-separated values)는 몇 가지 필드를 쉼표(,)로 구분하는 텍스트 데이터 및 텍스트 파일이다. 확장자는 .csv이며 MIME 형식은 text/csv이다. 오래전부터 스프레드시트나 데이터베이스 소프트웨어에서 많이 쓰였고, 2005년 10월 RFC 4180 (<https://tools.ietf.org/html/rfc4180>)에서 Informational로 사양이 문서화 되었다.



[그림 1] CSV 파일 형식 예시

2) SVG (Scalable Vector Graphics)

XML(Extensible Markup Language)형식으로 기술된 2차원 벡터 그래픽을 표현하기 위한 파일 형식이다. Chrome등의 웹 브라우저에서 바로 해석하여 렌더링이 가능하다. XML 형식으로 되어있으므로 메모장과 같은 문서 편집기로도 편집이 가능하다. 본 과제에서는 렌더링(rendering)이 제대로 수행되는지를 크롬 최신 버전에서 확인하여 평가한다. <[태그] {[어트리뷰트 이름]="[어트리뷰트 값]} x N(개)></[태그]> 형식으로 이루어진다. 또한 계층 구조를 가지며 <>와 </>사이에는 자식 노드들이 들어갈 수 있다.



[그림 2] SVG 파일 형식 예시

2. 프로그램 설명

본 프로젝트에서는 CSV Reader와 SVG Maker를 구현한다.

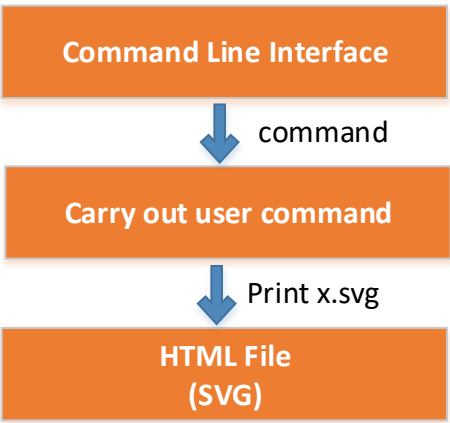
1) CSV Reader



[그림 3] CSV Reader의 동작 과정

- CSV Reader는 CSV 파일로부터 (,) 단위로 끊어서 입력을 받는다.
- CSV 파일은 SVG Maker에서 사용될 명령어에 대한 설명으로 이루어졌다.
- SVG Maker에서 사용할 help 명령어의 출력으로 사용되며, 또한 사용자로부터 잘못된 입력을 받았을 때, 오류 출력의 역할을 한다.
- CSV 파일의 파일명은 "help.csv"로 LMS의 과제게시판에 첨부되어있다.

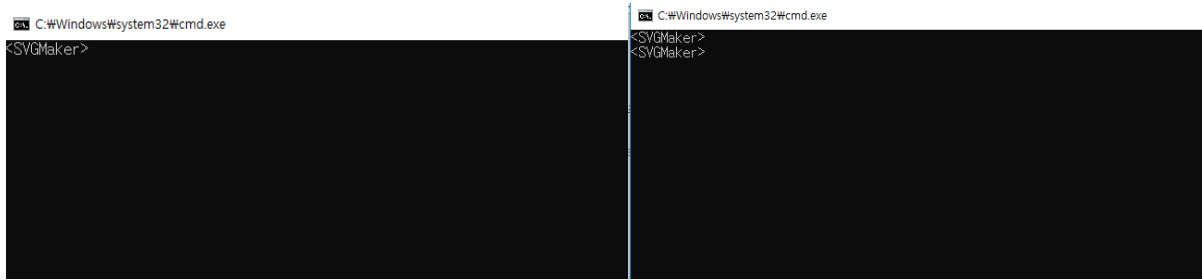
2) SVG Maker



[그림 4] SVG Maker의 동작 과정

- SVG Maker는 사용자의 명령을 입력 받아 최종적으로 HTML (SVG) 파일을 출력하는 프로그램이다.
- 사용자의 명령은 Standard input (표준 입력)으로 입력된다.
- [print] 명령을 통해 해당 명령 전까지의 SVG 파일을 출력한다.

- 초기 화면은 다음과 같이 구성된다.




[그림 4] SVG Maker의 초기화면

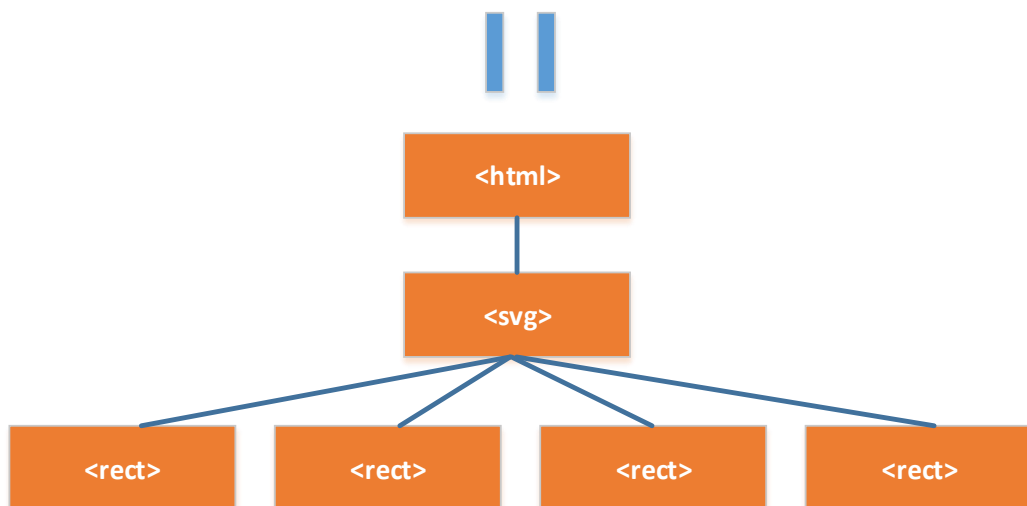
- <SVGMaker> 다음에 사용자로부터 명령어를 standard input(표준 입력)으로 받는다.
- 입력없이 enter를 누를 경우에 대해서도 [그림4-오른쪽]과 같이 처리해야함 (linux 터미널 CLI 환경처럼 동작)
- 아래 내용은 SVG Maker을 구성하기 위한 주요 요소이다.
 - ① SVG 객체
 - 본 프로젝트에서는 SVG 객체를 통해 SVG 파일 출력 전까지 HTML 트리를 구성해간다. (SVGObjectList 객체를 이용)
 - SVG 객체는 클래스로 선언되며, 명령에 따라 SVG 객체를 생성하거나 삭제한다.
 - SVG 객체는 tag name과 attribute 객체들을 포함한다. 예를 들어, <rect width="100" height="100"></rect>라는 SVG 객체의 tag name은 "rect"이며 2 개의 attribute 객체를 포함한다.
 - ② SVG Attribute 객체
 - Attribute 객체는 SVG 객체의 속성 값으로 attribute name과 attribute value의 쌍 <name, value>으로 구성된다.
 - SVG Attribute는 클래스로 구현한다.
 - 예를 들어 <rect width="100" height="100"></rect>의 경우 "rect" tag name의 SVG 객체는 <width, 100>과 <height, 100> 두 개의 attribute 객체를 갖는다.
 - SVG 객체마다 SVG Attribute 객체를 여러 개 가질 수 있고, 이를 SVGAttributeList 객체로 관리한다.

③ SVG 트리

- 본 프로젝트에서는 SVG Maker가 사용자의 명령을 입력 받아 SVG 객체를 트리 형태로 유지한다.
- SVG 트리의 구성 방법은 추후 클래스 설명에서 다룬다.
- SVG 트리의 예시는 [그림 5]와 같다.



```
svg_예시.html - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)
<html>
  <svg height="500" width="500">
    <rect fill="steelblue" height="10" transform="translate(0,0)" width="100"></rect>
    <rect fill="green" height="10" transform="translate(0,10)" width="200"></rect>
    <rect fill="orange" height="10" transform="translate(0,20)" width="300"></rect>
    <rect fill="red" height="10" transform="translate(0,30)" width="400"></rect>
  </svg>
</html>
```



[그림 5] SVG 트리 예시

④ Selection

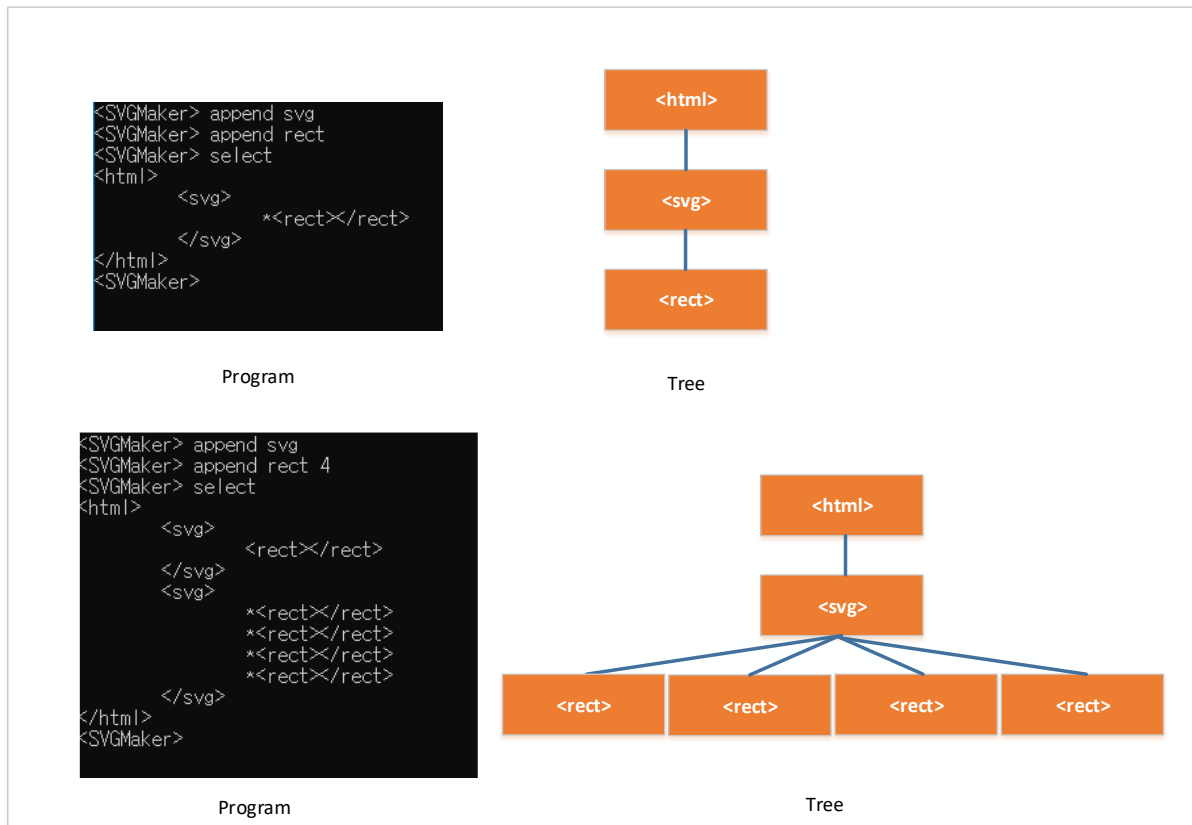
- User command를 수행할 때마다 SVG Maker가 선택하고 있는 SVG 객체가 달라 질 수 있다.
- SVG Maker가 선택하고 있는 SVG 객체들의 목록을 Selection이라고 한다.
- Selection에는 총 세 종류가 있으며 아래와 같다.
 - root-selection: root SVG 객체를 선택하고 있는 경우 (즉, <html> SVG 객체)
 - single-selection: root SVG 객체가 아닌 다른 SVG 객체 하나를 선택하고 있는 경우

- multiple-selection: 같은 부모를 둔 SVG 객체들을 선택하고 있는 경우
(multiple-selection이 0개 또는 1개의 SVG 객체를 선택하고 있을 수도 있다.)
- "html"을 태그 이름으로 하는 root SVG 객체는 프로그램 수행 시 자동으로 생성되어야 하며, SVG Maker의 초기 selection은 무조건 이 객체를 선택하는 root-selection이다.
- Selection은 클래스로 구현한다.
- Multiple-selection인 경우 Selection 객체들을 SelectList 객체로 관리한다.
- Selection 객체 하나 당 하나의 SVG 객체를 담당함 (만약 현재 Selection이 3개의 SVG 객체를 선택하고 있으면, Selection 객체는 총 3개가 SelectList 객체에 포함된다)

⑤ User Command (명령어+<argument>+[옵션]) – 총 9개

A. append

- 명령어 사용법: append <svg-tag-name> [n]
- Selection의 이동: (root-selection || single-selection) -> (single-selection || multiple-selection)
- 현재 Selection의 child로 <svg-tag-name>의 tag name을 가진 새로운 SVG 객체를 추가한다.
- 만약 [n] 옵션에 숫자가 들어가면, 현재 Selection의 child로 <svg-tag-name>의 tag name을 가진 [n] 개의 새로운 SVG 객체를 추가한다.
- 현재 Selection이 root-selection이나 single-selection일 때만 가능하고, multiple-selection일 경우 에러를 출력한다.
- **Selection은 새로 생성된 SVG 객체로 이동한다.**
- [그림 6]은 해당 명령어의 예시이다. (명령어 select와 에러 출력 부분은 추후 설명)



[그림 6] append 명령어 예시

B. select

- 명령어 사용법: select [<svg-tag-name>] [n]
- Selection의 이동: (root-selection || single-selection) -> (single-selection || multiple-selection)
(select만 입력할 경우 selection의 이동 없음)
- 현재 Selection의 child 중에 <svg-tag-name>을 tag name으로 가지는 모든 객체로 Selection을 이동한다.
- 현재 selection이 하나의 SVG 객체만 가리킬 때만 가능하다.
- 만약 [n 옵션]에 숫자가 들어가면, 현재 Selection의 child 중에 <svg-tag-name>을 tag name으로 가지는 객체 중 [n] 번째 객체로 Selection을 이동한다.
즉, (root-selection || single-selection) -> single selection
- 만약 select만 입력한다면, 현재 SVG 트리를 출력하고 selection이 가리키는 SVG 객체 앞에 "*"를 붙여, 현재 selection의 상태를 알려준다.
- 현재 Selection이 root-selection이나 single-selection일 때만 가능하고, multiple-selection일

경우 에러를 출력한다.

- Selection이 이동한 후에는 몇 개의 item이 선택됐는지 출력한다. **child 중에 일치하는 tag 명을 가진 SVG 객체가 없다면**, 에러를 출력하고 현재 Selection을 유지한다.
- [그림 7]은 해당 명령어의 예시이다. (에러 출력 부분은 추후 설명)

```
<SVGMaker> append svg
<SVGMaker> append rect 4
<SVGMaker> end
<SVGMaker> select
<html>
  *<svg>
    <rect></rect>
    <rect></rect>
    <rect></rect>
    <rect></rect>
  </svg>
</html>
<SVGMaker> select rect
4 item(s) is selected!
<SVGMaker> select
<html>
  <svg>
    *<rect></rect>
    *<rect></rect>
    *<rect></rect>
    *<rect></rect>
  </svg>
</html>
<SVGMaker>
```

Example 1

```
<SVGMaker> select
<html>
  *<svg>
    <rect></rect>
    <rect></rect>
    <rect></rect>
    <rect></rect>
  </svg>
</html>
<SVGMaker> select rect 3
1 item(s) is selected!
<SVGMaker> select
<html>
  <svg>
    <rect></rect>
    <rect></rect>
    *<rect></rect>
    <rect></rect>
  </svg>
</html>
<SVGMaker>
```

Example 2

[그림 7] select 명령어 예시

C. remove

- 명령어 사용법: remove
- Selection의 이동: (single-selection || multiple-selection) -> (single-selection || root-selection)
- 현재 selection이 선택하고 있는 모든 SVG 객체를 삭제한다
- 명령 수행 후에는 삭제된 SVG 객체(들)의 부모를 선택한다.
- 만약 root-selection일 경우, 에러를 출력한다.
- SVG 객체를 삭제할 때는 삭제되는 SVG 객체들의 child들도 함께 삭제한다.

- [그림 8]은 해당 명령어의 예시이다. (에러 출력 부분은 추후 설명)

```

<SVGMaker> select
<html>
  *<svg>
    <rectX/rect>
    <rectX/rect>
    <rectX/rect>
    <rectX/rect>
    <rectX/rect>
  </svg>
  <svg>
    <circleX/circle>
    <circleX/circle>
    <circleX/circle>
  </svg>
</html>
<SVGMaker> remove
<SVGMaker> select
*<html>
  <svg>
    <circleX/circle>
    <circleX/circle>
    <circleX/circle>
  </svg>
</html>

```

Example 1

[그림 8] remove 명령어 예시

D. end

- 명령어 사용법: end
- Selection의 이동: (single-selection || multiple-selection || root-selection) -> (single-selection || root-selection)
- 현재 selection (들)의 부모 노드를 선택한다.
- 만약 **root-selection**일 경우, 에러를 출력한다.
- [그림 9]는 해당 명령어의 예시이다. (에러 출력 부분은 추후 설명)

```

<SVGMaker> append svg
<SVGMaker> append rect 5
<SVGMaker> select
<html>
    <svg>
        *<rect></rect>
        *<rect></rect>
        *<rect></rect>
        *<rect></rect>
        *<rect></rect>
    </svg>
</html>
<SVGMaker> end
<SVGMaker> select
<html>
    *<svg>
        <rect></rect>
        <rect></rect>
        <rect></rect>
        <rect></rect>
        <rect></rect>
    </svg>
</html>
<SVGMaker>

```

Example 1

[그림 9] end 명령어 예시

E. exit

- 명령어 사용법: exit
- Selection의 이동: None
- 프로그램을 바로 종료한다.

F. cattr

- 명령어 사용법: cattr <svg-attr-name> <svg-attr-value>
- Selection의 이동: None
- 현재 Selection의 SVG 객체(들)의 attribute를 추가하거나 수정한다.
- 만약 <svg-attr-name>과 일치하는 attribute name이 없다면, 해당 <svg-attr-name>과 <svg-attr-value>를 가지는 SVG Attribute 객체를 생성하여 추가한다.

- [그림 10]은 해당 명령어의 예시이다. (에러 출력 부분은 추후 설명)

```
<SVGMaker> append svg
<SVGMaker> select
<html>
  *<svg></svg>
</html>
<SVGMaker> cattr width 500
<SVGMaker> cattr height 500
<SVGMaker> select
<html>
  *<svg width="500" height="500"></svg>
</html>
<SVGMaker> cattr width 1000
<SVGMaker> select
<html>
  *<svg width="1000" height="500"></svg>
</html>
```

Example 1

```
<SVGMaker> append svg
<SVGMaker> append rect 4
<SVGMaker> select
<html>
  <svg>
    *<rect></rect>
    *<rect></rect>
    *<rect></rect>
    *<rect></rect>
  </svg>
</html>
<SVGMaker> cattr width 500
<SVGMaker> cattr height 500
<SVGMaker> select
<html>
  <svg>
    *<rect width="500" height="500"></rect>
    *<rect width="500" height="500"></rect>
    *<rect width="500" height="500"></rect>
    *<rect width="500" height="500"></rect>
  </svg>
</html>
<SVGMaker> end
<SVGMaker> select rect 2
1 item(s) is selected!
<SVGMaker> cattr width 250
<SVGMaker> select
<html>
  <svg>
    <rect width="500" height="500"></rect>
    *<rect width="250" height="500"></rect>
    <rect width="500" height="500"></rect>
    <rect width="500" height="500"></rect>
  </svg>
</html>
<SVGMaker>
```

Example 2

[그림 10] cattr 명령어 예시

G. tattr

- 명령어 사용법: `tattr <x-multiplier> <y-multiplier>`
- Selection의 이동: None
- 현재 Selection의 SVG 객체(들)을 translate한다.
- SVG 객체의 translate는 해당 객체에 `transform="translate(x,y)"` 형식의 attribute를 넣음으로써 수행할 수 있다. (x, y는 소수점 6번째 자리까지 표시)
- 이때, x와 y에는 `<_multiplier> * index`에 해당하는 수치가 들어간다. (index는 현재 selection 중 해당 SVG객체가 몇 번째에 위치하는지에 대한 순서 정보이다.)
- 만약 이미 translate에 해당하는 attribute가 있으면, 현재 명령어로 할당되는 값으로 수정한다.
- [그림 11]은 해당 명령어의 예시이다. (에러 출력 부분은 추후 설명)

```
<SVGMaker> select
<html>
  <svg>
    *<rect width="500" height="500"></rect>
    *<rect width="250" height="500"></rect>
    *<rect width="500" height="500"></rect>
    *<rect width="500" height="500"></rect>
  </svg>
</html>
<SVGMaker> tattr 20 30
<SVGMaker> select
<html>
  <svg>
    *<rect width="500" height="500" transform="translate(0.000000,0.000000)"></rect>
    *<rect width="250" height="500" transform="translate(20.000000,30.000000)"></rect>
    *<rect width="500" height="500" transform="translate(40.000000,60.000000)"></rect>
    *<rect width="500" height="500" transform="translate(60.000000,90.000000)"></rect>
  </svg>
</html>
<SVGMaker>
```

Example 1

[그림 11] tattr 명령어 예시

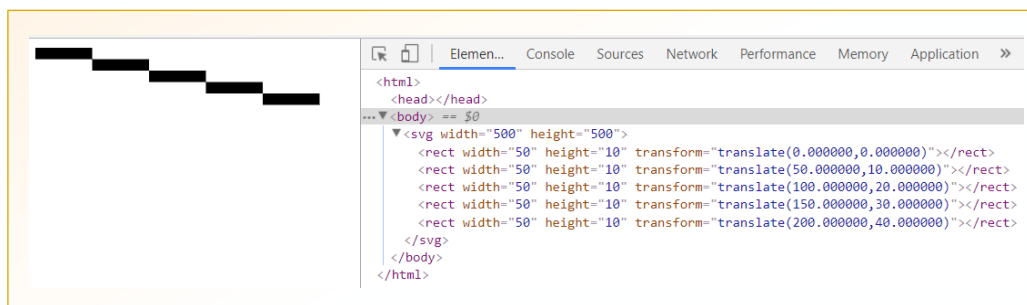
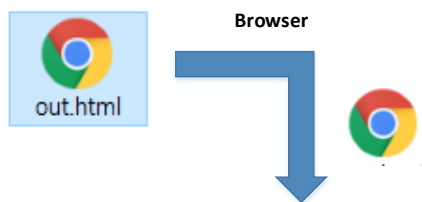
H. print

- 명령어 사용법: `print [output-file-name]`
- Selection의 이동: None
- 현재까지 작성된 HTML 문서를 [output-file-name] 이름을 가진 파일에 출력한다.
- 만약 "print"만 입력하면, HTML 문서를 standard output(표준 출력)으로 출력한다.

- 출력 양식은 아래의 양식을 재귀적으로 호출하면 된다.
 - `<tag attr1="attr1-value" attr2="attr2-value">`
(tag의 자식들 출력)
`</tag>`
- Root SVG 객체는 탭("Wt")가 앞에 0개 붙고, 자식 SVG 객체는 부모 SVG 객체보다 탭("Wt")가 하나씩 더 붙는다.
- [그림 x]는 해당 명령어의 예시이다. (에러 출력 부분은 추후 설명)
 - `<head>`와 `<body>` object는 browser가 자동적으로 붙여준다.
 - 자식이 있는 SVG 객체일 경우 `</(svg-tag-name)>`의 값을 자식들을 모두 출력한 후 붙여준다.
 - [그림 12]와 동일한 양식처럼 html 파일이 출력되게 해야합니다.

```
<SVGMaker> print
<html>
  <t  <svg width="500" height="500">
    <t  <rect width="50" height="10" transform="translate(0.000000,0.000000)"></rect>
    <rect width="50" height="10" transform="translate(50.000000,10.000000)"></rect>
    <rect width="50" height="10" transform="translate(100.000000,20.000000)"></rect>
    <rect width="50" height="10" transform="translate(150.000000,30.000000)"></rect>
    <rect width="50" height="10" transform="translate(200.000000,40.000000)"></rect>
  </svg>
</html>
<SVGMaker> print out.html
"out.html" file is created
```

Example 1



[그림 12] print 명령어 예시

I. help

- 명령어 사용법: help [output-file-name]
- Selection의 이동: None
- 만약 help만 입력하면, "help.csv"파일의 모든 command의 usage 부분만 모두 출력한다.
- 만약 help [command-name]을 입력하면, [command-name]에 해당하는 command의 모든 필드를 출력한다. 또한 [그림 14]와 같이 명령어 오류(명령어는 있지만, 명령이 틀린 경우)에 대한 출력을 이와 같이 해준다.
 - [COMMAND] "help.csv"의 [command-name]의 command
 - [USAGE] "help.csv"의 [command-name]의 usage
 - [CONDITION] "help.csv"의 [command-name]의 condition
 - [DESCRIPTION] "help.csv"의 [command-name]의 description
- [그림 13]는 해당 명령어의 예시이다.

```
<SVGMaker> help
append <svg-tag-name> [n]
select [<svg-tag-name>] [n]
remove
end
exit
cattr <svg-attr-name> <svg-attr-value>
tattr <x-multiplier> <y-multiplier>
print [output-file-name]
help [command-name]
<SVGMaker> help cattr
[COMMAND] cattr
[USAGE] cattr <svg-attr-name> <svg-attr-value>
[CONDITION] None
[DESCRIPTION] Add or create an attribute named <svg-attr-name> and having the <svg-attr-value> to the SVG Object(s) of
the current selection
<SVGMaker>
```

☐ Example 1

	A	B	C	D
1	command	usage	condition	description
2	append	append <svg-tag-name> [n]	(root-selection single-selection) -> (single-selection multiple-selection)	Add a new S
3	select	select [<svg-tag-name>] [n]	(root-selection single-selection) -> (single-selection multiple-selection)	Select one S
4	remove	remove	(single-selection multiple-selection) -> (single-selection root-selection)	Remove all s
5	end	end	(single-selection multiple-selection root-selection) -> (single-selection root-selection)	Select the p
6	exit	exit	None	Exit this pro
7	cattr	cattr <svg-attr-name> <svg-attr-value>	None	Add or creat
8	tattr	tattr <x-multiplier> <y-multiplier>	None	Translate the
9	print	print [output-file-name]	None	Print a HTML
10	help	help [command-name]	none	Print [comm

☐ help.csv

[그림 13] help 명령어 예시

```

<SVGMaker> append svg 5 circle
[Error] Wrong command!
[COMMAND] append
[USAGE] append <svg-tag-name> [n]
[CONDITION] (root-selection || single-selection) -> (single-selection || multiple-selection)
[DESCRIPTION] Add a new SVG object named <svg-tag-name> tag
<SVGMaker> end append
[Error] Wrong command!
[COMMAND] end
[USAGE] end
[CONDITION] (single-selection || multiple-selection || root-selection) -> (single-selection || root-selection)
[DESCRIPTION] Select the parent SVG object of the current selection
<SVGMaker>

```

[그림 14] 명령어 에러 출력 예시

3. 클래스 설명

- SVGObject, SVGAttribute, Selection, SVGObjectList, SVGAttributeList, SelectionList 클래스를 필수로 요구한다.
- 클래스 추가가 있으면, 그에 대한 이유를 보고서에 명시해야한다. 동일한 클래스의 인스턴스들은 서로 링크드 리스트 형태로 연결된다.
- 링크드 리스트 사용을 위해서, 연결 노드를 가리키는 포인터 변수와 그 변수를 다루는 함수 추가해야한다. 즉, [그림 x]의 클래스 다이어그램에는 해당 내용을 포함하지 않는다.
(STL, Boost와 같은 라이브러리의 구현된 리스트를 사용하는 것을 금지한다.)
- 또한, "print" 명령어와 "select" 명령어를 위한 출력 관련 함수 추가를 허용한다.
(출력 방법에 대한 내용을 보고서에 상세히 기입해야한다.)
- 그 외 필요한 멤버 변수나 멤버 함수 추가도 허용하지만, 이 경우 반드시 보고서에 그 이유를 명시해야한다.
- 멤버 변수에 접근하기 위한 **생성자(오버로딩)**, **소멸자(오버로딩)**, **get 멤버 함수**, **set 멤버 함수**의 경우 자유롭게 구성(추가/삭제)해도 무관하다.
- CSV Reader의 경우 자유롭게 구현하며 (단일 함수 혹은 클래스 혹은 메인 함수 내), **CSV Reader의 구현 방법에 대한 설명을 보고서에 명시해야한다.**

1) 필수 클래스(6개)의 클래스 다이어그램 및 설명

SVGObject	SVGAttribute	Selection
-aTagName: String -apParent: SVGObject* -apChildList: SVGObjectList* -apAttributeList: SVGAttributeList* +SVGObject() +~SVGObject() +AppendChild(pChild: SVGObject*) +AddAttribute(pAttribute: SVGAttribute*) +SetTagName(tagName: string) +GetTagName(): string	-aName: String -aValue: String +SVGAttribute() +~SVGAttribute() +SetName(name: string) +GetName(): string	-apSvgObject: SVGObject* +Selection() +~Selection() +SetSvgObject(pSvgObject: string) +GetSvgObject(): string
SVGObjectList	SVGAttributeList	SelectionList
-apHead: SVGObject* -aSize: int +SVGObjectList() +~SVGObjectList() +AddSvgObject(pObject: SVGObject*) +RemoveSvgObject(pObject: SVGObject*) +SetHead(pHead: SVGObject*) +GetHead(): SVGObject*	-apHead: SVGAttribute* -aSize: int +SVGAttributeList() +~SVGAttributeList() +AddSvgAttribute(pAttribute: SVGAttribute*) +SetHead(pHead: SVGAttribute*) +GetHead(): SVGAttribute*	-apHead: Selection* -aSize: int +SelectionList() +~SelectionList() +AddSelection(pSelection: Selection*) +RemoveSelection(pSelection: Selection*) +Clear() +SetHead(pHead: Selection*) +GetHead(): Selection*

[그림 14] 필수 클래스의 클래스 다이어그램

① SVGObject

멤버 변수	멤버 함수
aTagName: SVG객체의 태그명 apParent: 부모 SVG 객체 포인터 apChildList: 자식 SVG 객체(들)에 대한 리스트 포인터 apAttributeList: SVG 객체의 Attribute(들)의 리스트 포인터	AppendChild(pChild: SVGObject*) : 자식 SVG 객체를 추가함 AddAttribute(pAttribute: SVGAttribute*) : SVG Attribute를 추가함

② SVGAttribute

멤버 변수	멤버 함수
aName: Attribute 명 aValue: Attribute 값

③ Selection

멤버 변수	멤버 함수
apSVGObject: 현재 Selection이 가리키고있는 SVG 객체 포인터

④ SVGObjectList

멤버 변수	멤버 함수
apHead: 현재 리스트의 head 포인터 aSize: 현재 리스트의 element 수	AddSvgObject(pObject: SVGObject*) : 리스트에 pObject를 추가함 RemoveSvgObject(pObject: SVGObject*) : 리스트에서 pObject를 제거함

⑤ SVGAttributeList

멤버 변수	멤버 함수
apHead: 현재 리스트의 head 포인터 aSize: 현재 리스트의 element 수	AddSvgAttribute(pAttribute: SVGAttribute*) : 리스트에 pAttribute를 추가함

⑥ SelectList

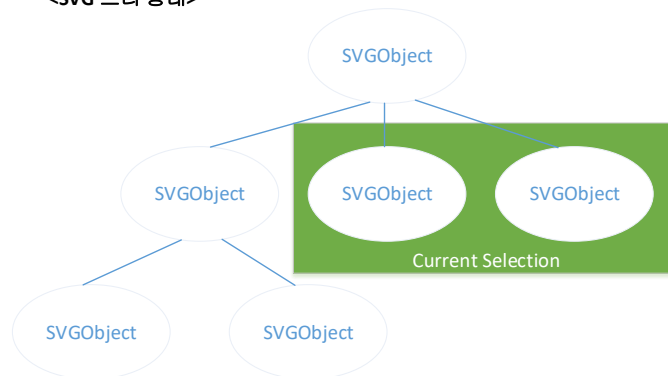
멤버 변수	멤버 함수
<p>apHead: 현재 리스트의 head 포인터</p> <p>aSize: 현재 리스트의 element 수</p> <p>.....</p>	<p>AddSelection(pSelection: SVGObject*)</p> <p>: 리스트에 pSelection을 추가함</p> <p>RemoveSelection(pSelection: SVGObject*)</p> <p>: 리스트에서 pSelection을 제거함</p> <p>Clear()</p> <p>: 현재 리스트를 비움 (apHead = NULL / aSize = 0)</p> <p>모든 현재 리스트의 Selection instance들을 할당 해제</p> <p>.....</p>

18

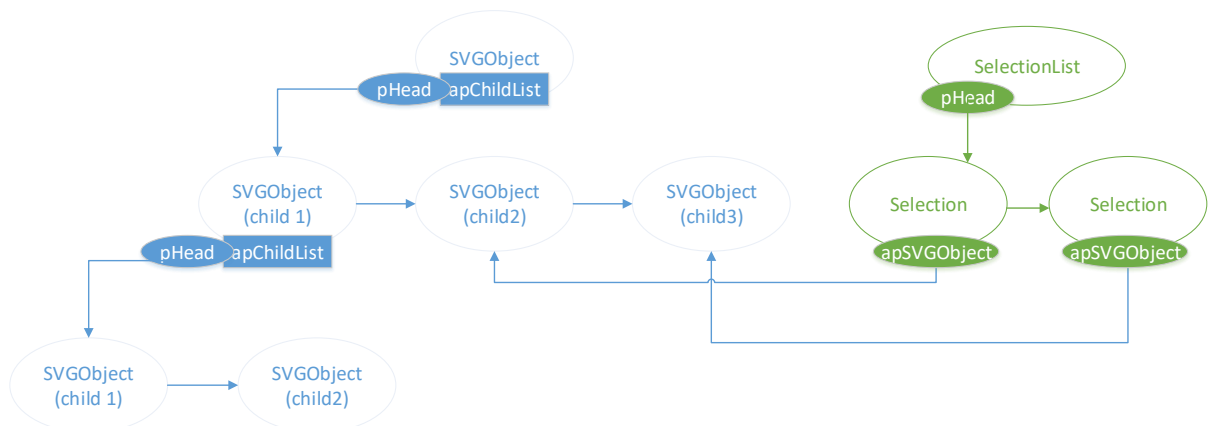
※ 모든 클래스의 생성자(오버로딩), 소멸자(오버로딩), get 멤버 함수, set 멤버 함수는 자유롭게 사용

2) SVG 트리와 Selection 예시 (링크드 리스트 사용)

<SVG 트리 형태>



<링크드리스트로 구현>

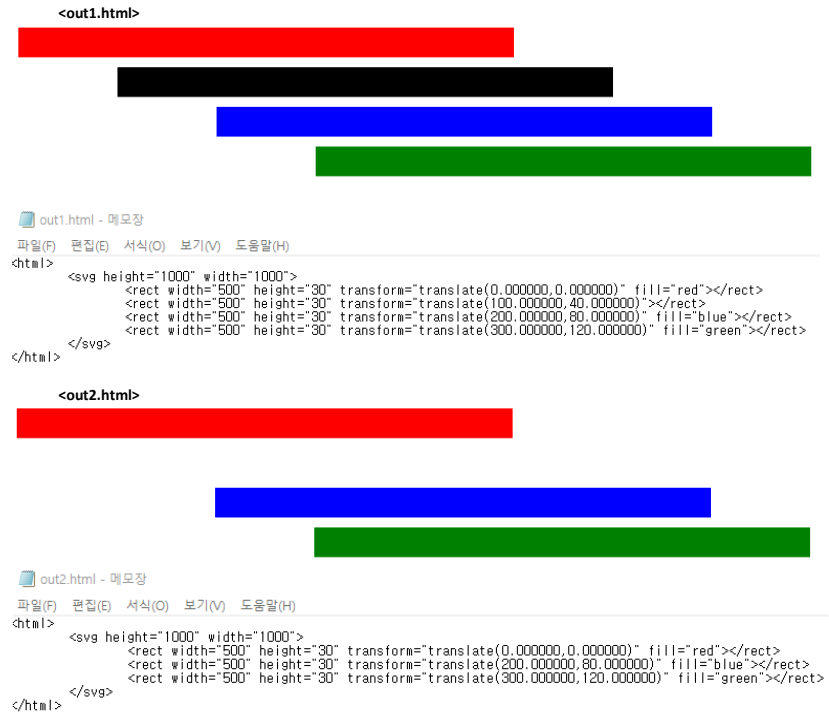


[그림 16] SVG 트리와 Selection 예시

4. 프로그램 예시

1) [예시 1]

```
<SVGMaker> select
<html></html>
<SVGMaker> append svg
<SVGMaker> cattr height 1000
<SVGMaker> cattr width 1000
<SVGMaker> append rect 4
<SVGMaker> end
<SVGMaker> select rect
4 item(s) is selected!
<SVGMaker> cattr width 500
<SVGMaker> cattr height 30
<SVGMaker> end
<SVGMaker> select rect
4 item(s) is selected!
<SVGMaker> tattr 100 40
<SVGMaker> end
<SVGMaker> select rect 1
1 item(s) is selected!
<SVGMaker> cattr fill red
<SVGMaker> end
<SVGMaker> select rect 3
1 item(s) is selected!
<SVGMaker> cattr fill blue
<SVGMaker> end
<SVGMaker> select rect 4
1 item(s) is selected!
<SVGMaker> cattr fill green
<SVGMaker> print out1.html
"out1.html" file is created
<SVGMaker> end
<SVGMaker> select rect 2
1 item(s) is selected!
<SVGMaker> remove
<SVGMaker> print out2.html
"out2.html" file is created
<SVGMaker>
```



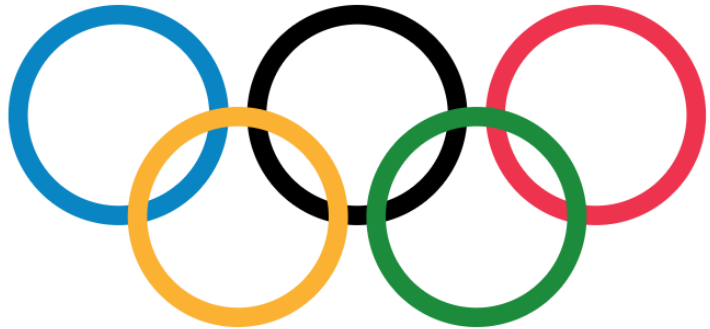
[그림 17] 프로그램 예시 1

2) [예시 2]

20

```
<SVGMaker> append svg
<SVGMaker> cattr width 900
<SVGMaker> cattr height 900
<SVGMaker> append rect
<SVGMaker> end
<SVGMaker> append g
<SVGMaker> append circle 5
<SVGMaker> end
<SVGMaker> end
<SVGMaker> select rect
1 item(s) is selected!
<SVGMaker> cattr width 900
<SVGMaker> cattr height 600
<SVGMaker> cattr style fill:#fff
<SVGMaker> end
<SVGMaker> select g
1 item(s) is selected!
<SVGMaker> cattr stroke-width 17.67258644
<SVGMaker> cattr stroke #000
<SVGMaker> cattr fill none
<SVGMaker> select circle 1
1 item(s) is selected!
<SVGMaker> cattr cx 232.34814
<SVGMaker> cattr cy 253.49319
<SVGMaker> cattr r 91.618408
<SVGMaker> cattr stroke #0885c2
<SVGMaker> end
<SVGMaker> select circle 2
1 item(s) is selected!
<SVGMaker> cattr cx 450
<SVGMaker> cattr cy 253.49319
<SVGMaker> cattr r 91.618408
<SVGMaker> end
<SVGMaker> select circle 3
1 item(s) is selected!
<SVGMaker> cattr cx 667.65186
<SVGMaker> cattr cy 253.49319
<SVGMaker> cattr r 91.618408
<SVGMaker> cattr stroke #ed334e
<SVGMaker> end
<SVGMaker> select circle 4
1 item(s) is selected!
<SVGMaker> cattr cx 341.17407
<SVGMaker> cattr cy 346.50681
<SVGMaker> cattr r 91.618408
<SVGMaker> cattr stroke #fbb132
<SVGMaker> end
<SVGMaker> select circle 5
1 item(s) is selected!
<SVGMaker> cattr cx 558.82593
<SVGMaker> cattr cy 346.50681
<SVGMaker> cattr r 91.618408
<SVGMaker> cattr stroke #1c8b3c
<SVGMaker> end
<SVGMaker> print out.html
"out.html" file is created
<SVGMaker>
<SVGMaker> select circle 2
1 item(s) is selected!
<SVGMaker> remove
<SVGMaker> print out2.html
"out2.html" file is created
<SVGMaker>
```

<out.html>

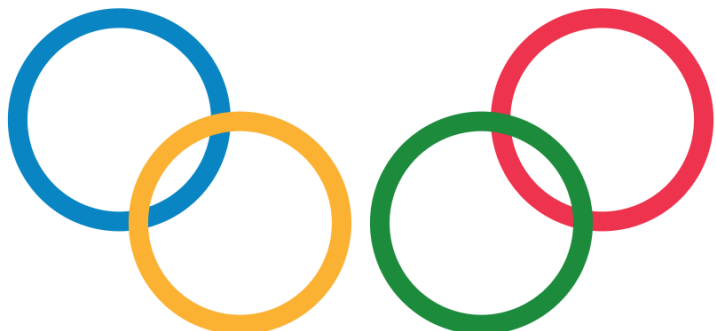


out.html - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
<html>
<svg width="900" height="900">
  <rect width="900" height="600" style="fill:#fff"></rect>
  <g stroke-width="17.67258644" stroke="#000" fill="none">
    <circle cx="232.34814" cy="253.49319" r="91.618408" stroke="#0885c2"></circle>
    <circle cx="450" cy="253.49319" r="91.618408"></circle>
    <circle cx="667.65186" cy="253.49319" r="91.618408" stroke="#ed334e"></circle>
    <circle cx="341.17407" cy="346.50681" r="91.618408" stroke="#fbb132"></circle>
    <circle cx="558.82593" cy="346.50681" r="91.618408" stroke="#1c8b3c"></circle>
  </g>
</svg>
</html>
```

<out2.html>



out2.html - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
<html>
<svg width="900" height="900">
  <rect width="900" height="600" style="fill:#fff"></rect>
  <g stroke-width="17.67258644" stroke="#000" fill="none">
    <circle cx="232.34814" cy="253.49319" r="91.618408" stroke="#0885c2"></circle>
    <circle cx="667.65186" cy="253.49319" r="91.618408" stroke="#ed334e"></circle>
    <circle cx="341.17407" cy="346.50681" r="91.618408" stroke="#fbb132"></circle>
    <circle cx="558.82593" cy="346.50681" r="91.618408" stroke="#1c8b3c"></circle>
  </g>
</svg>
</html>
```

[그림 18] 프로그램 예시 2

Requirements

- 본 과제의 프로그램은 리눅스 환경(프로그래밍 실습 서버 또는 MinGW 또는 맥에서의 터미널 등 그와 유사한 환경)에서 컴파일 및 구동 가능하도록 작성한다.
- 본 과제의 소스코드는 다음과 같이 13개의 파일로 구성하기를 권장한다. 다른 형태의 파일 구성도 허용하지만, 그에 대한 이유를 보고서에 명시하여야 한다.
 - 메인함수를 포함하는 .cpp 파일
 - 각 객체 클래스(SVGObject, SVGAttribute, Selection)의 선언 및 정의에 대한 .cpp 및 .h 파일
 - 각 리스트 클래스(SVGObjectList, SVGAttributeList, SelectionList)의 선언 및 정의를 위한 .cpp 및 .h 파일
- 반드시 Makefile을 작성하여 컴파일이 가능하도록 해야 하며, 작성된 Makefile 역시 제출 해야 한다. (make를 통한 컴파일 실패 시, 컴파일이 안되는 것으로 간주)
- 실습 서버의 접속 방법은 첨부된 [프로그래밍 실습환경 사용가이드]의 '실습 시스템 접속 방법'을 참조한다.
- 상기한 환경에서의 프로그램 컴파일 및 실행 과정에 대한 설명과 이미지 모두 보고서에 포함해야 한다.
- SVG 객체, SVG Attribute 객체, Selection 객체, 각 객체의 링크드리스트는 모두 클래스로 구현하여야 한다. (클래스 미사용 시 0점)
- 명령에 수행에 대해 각자의 테스트 명령어를 통해 확인하기 바랍니다. (조교가 채점 시 조교 테스트 코드로 명령어 수행에 대한 결과를 확인할 것임)
- 상속(Inheritance)은 허용하지만, 이에 대한 설명을 보고서에 기입하기 바랍니다. (필수 클래스 6개는 지켜주시기 바랍니다.)
- STL이나 Boost 등의 라이브러리는 사용하지 않는다. (사용할 경우 0점)
- 구성원들의 모든 멤버 변수는 'private'이어야 하며, 이들 멤버 변수에 대한 접근은 모두 'public' 멤버 함수들을 통해서만 이루어져야 한다.
- 위에 설명된 프로그램의 형식을 모두 만족하여야 한다.
- 프로그램은 사용자가 종료("exit")를 하기 전까지 반복 실행되어야 한다.
- **Output file (HTML 파일)이 생성되면 GUI 환경 (Linux 환경의 GUI or Windows 환경의 GUI)에서 최신 크롬 브라우저로 실행하여 output을 확인하기 바랍니다.**

- 모든 예외 상황(잘못된 명령, 파일 열기 실패, 잘못된 Selection 조건, 객체 수정 또는 삭제 시 객체가 존재하지 않는 경우 등)에 대해 적절한 에러 메시지를 출력해야 한다.
- 보고서는 사용된 클래스의 멤버 변수/함수에 대한 설명을 포함하여야 한다.
- 보고서는 위의 모든 요구조건을 확인할 수 있는 시나리오와 그 결과를 캡처한 그림을 포함하여야 한다. (+프로그램 [예시 1]과 [예시 2] 에서 제시한 명령어 수행에 대한 결과)
- 보고서에 위 요구사항 중 만족한 것과 만족하지 못한 것을 정확하게 명시하여야 하며, 정확하게 명시하지 않을 경우 '프로그램 설계 및 구현', '보고서 구성 및 내용, 양식' 점수가 감점될 수 있다.
- 보고서 및 소스코드는 LMS의 과제 디렉토리에 업로드한다.
- 채점 기준은 AssnReadMe_Spring_2018.pdf 파일을 참고한다.