2023 Spring OOP Assignment Report

과제 번호 : 3
학번 : 20220302
이름 : 김지현
Povis ID : jihyunk

명예서약 (Honor Code)

나는 이 프로그래밍 과제를 다른 사람의 부적절한 도움 없이 완수하였습니다.
I completed this programming task without the improper help of others.

1. 프로그램 개요

- 이번 project 는 주어진 txt 파일을 이용해 ASCII art 를 만드는 것이다.
- ASCII art 란 한 그림을 픽셀화 시킨 후 각 픽셀의 명도에 따라 값을 저장한 뒤 (예를 들면 어두운 픽셀칸은 큰 값을, 밝은 픽셀 칸은 작은 값을 가지도록 한다) 복잡한 문자와 단순한 문자를 이용해 그림을 구현하는 것이다.
- ASCII art 는 세 가지 class 를 이용할 것이고 각각의 class 는 데이터 타입을 읽어오고, 숫자 값을 저장하고, 마지막으로 문자로 변환하는 역할을 할 것이다.
- 프로그램을 실행하기 위해서는 project 탭에서 project property 에 들어 간 뒤, debugging 칸에 있는 command argument 에 세가지 txt 파일의 이름을 입력해야 한다. 첫번째 txt는 픽셀의 width, height 과 각 픽셀 값이 '|' 를 이용해 나눠져 있는 파일이다. 두번째 txt 는 configuration 을 말해주는 파일으로 후에 나올 artist 와 draw 방식을 저장한다. 마지막 txt 는 결과 그림을 저장할 파일이다.

2. 프로그램의 구조 및 알고리즘

parser	
private	Public
	Vector<int> load_image (const char*) Vector <string> load_config (const char*) Void write_result(const char*, const string&)

- Load_image 는 char* 로 파일 이름을 받아온 후 <fstream> 헤더를 이용해 파일을 연다. 전체 파일에 입력 되어 있는 픽셀값들과 '|' 를 하나의 string 변수 whole_text 로 저장한 뒤, for 문 속에서 char temp 를 이용해 문자 하나씩 확인할 것이다. 만약

temp 에 저장된 값이 '|' 가 아니라면 number_str 이라는 string 변수에 값을 더해주고 '|' 가 맞다면 number_str 에 저장된 string 값을 stoi 함수를 이용해 integer 로 바꿔준다. 위에서 선언해둔 pixel 이라는 integer 타입의 vector 에 push_back 을 이용해 integer 값을 추가해주고 number_str 은 초기화해준다. 만약 temp 가 whole_text 에 있는 모든 문자를 확인했다면 number_str 에 저장되어 있는 string 값을 integer로 변환한 후 vector 에 추가해준다. 마지막으로 앞에서 열었던 파일을 닫은 후 vector<int> 로 저장한 pixel 이라는 vector 를 반환해준다.

- Load_config 는 위에서 설명한 load_image 랑 유사하게 입력된 파일에서 값을 가져오게 된다. 앞에서는 integer 값을 가져온 반면, 이번엔 string 타입을 가지는 vector 변수 config 를 선언한다. 위와 같은 방식으로 파일을 열고 값을 읽어 온 뒤, '|' 가 나올 때 마다 임시 string 변수인 class_str 에 저장된 값을 vector config 에 추가해준다. 이때, char 변수가 전체 파일 데이터를 하나씩 읽어 올 때, string 에 char 를 추가하기 위해 append 함수를 사용했다. 모든 값을 다 읽어 왔다면 class_str 에 남아있는 값을 마지막으로 vector 에 추가해주고 파일을 닫는다. 그리고 config 라는 vector 를 반환해준다.

- 마지막 멤버 함수인 write_result 는 output.txt 파일에 결과 값을 저장하는 것이다. 이 함수의 definition은 assn 3 pdf 에 주어졌다.

Artist	
private	public
Int width Int height Vector <int> numbers Vector <char> characters	Artist(int, int, const vector<int>&) Virtual char mapper(int, int) Void setchar(vector<char>) Int getint (int, int) Vector<char> getchar() Int getwidth() Int getheigt()

- 메인 함수에서 새로운 artist 가 선언 될 때, width, height, 그리고 load_image 에서 얻은 pixel 값을 받아오게 된다. Constructor 를 이용해 각 값을 private 변수로 저장해준다.

- Vector <char> character는 받아온 integer vector를 이용해 mapper 에서 값이 int 에서 char 로 변활 될 때 저장 될 vector 이다.

- Load_config 에서 classic, iclassic, soblex, sobley, 그리고 gradient 중 하나가 입력 받아질 것이다. 각 category 에 따라 mapper 함수의 definition이 달라지기 때문에

virtual 로 선언 한 뒤 각 category 의 class 에서 definition 을 선언해줄 것이다.

- Void setchar 함수는 각 category 에서는 artist class 의 private 멤버에 접근할 수 없기 때문에 character vector 를 저장할 때 사용되는 setter 함수이다.
- Int getint 는 2차원 픽셀값의 x, y 좌표를 받아왔을 때, 1차원 vector 에 저장되어 있는 값을 반환해주는 함수이다. 이 함수에서의 position 은 2차원 좌표를 1차원으로 바꾸는 역할을 한다. Y 좌표 * width + x 좌표 를 이용하면 position 을 얻을 수 있다.
- Class artist 의 private 멤버는 다른 class 에서의 접근이 어렵기 때문에 각각의 category 에서 private 멤버 변수를 사용하기 위해 getchar, getwidth, getheight 함수를 이용해 각 값을 반환한 뒤, 다른 class 에서 선언된 변수에 그 값을 저장하도록 했다.
- 밑의 class 들은 artist class 의 public 멤버를 상속받은 것이다.

Classic (artist 상속)	
private	Public
	Classic (int width, int height, cost vector<int>& pixel) : artist (width, height, pixel) Char mapper (int, int)

- Classic 멤버 함수는 constructor 이다. 메인 함수에서 width, height, 그리고 각 픽셀의 값을 받아오면 artist 의 constructor 를 불러와 artist 의 private 멤버에 각 value 를 저장해준다.
- Constructor 를 이용해 값을 저장함과 동시에 mapper 함수를 이용해 integer 타입으로 받아온 vector를 character 타입으로 바꿔준다. Classic class 는 값에 따라 복잡한 문자부터 단순한 문자를 할당하기 때문에 두 개의 for 문을 이용해서 2차원 배열처럼 사용할 것이다.
- Char Mapper (int, int) 에서는 각 x, y 좌표를 받아온 후 artist class 의 int get int 를 이용해 좌표 값을 받아온 뒤, 17로 나눠준 몫을 integer 타입으로 quotient 변수에 저장해준다. 그리고 switch 문을 이용해 각 문자를 반환한다.
- 이렇게 반환된 문자는 class 가 construct 될 때 생성한 vector <char> chars 라는 벡터에 추가해 준다.
- 두 개의 for 문이 끝나고 나면 문자 가 저장된 chars 변수를 artist class 의 set char 멤버 함수의 argument 로 넘겨준다.

Iclassic (artist 상속)	
private	public
	Iclassic (int width, int height, cost vector<int>& pixel) : artist (width, height, pixel) Char mapper (int, int)

□ Iclassic class 는 classic 과 같은 방식으로 실행 되지만 mapper 에서 작은 숫자가 단순한 문자를 반환받고 큰 숫자는 복잡한 문자를 반환받는다.

Soblex (artist 상속)	
private	public
	soblex (int width, int height, cost vector<int>& pixel) : artist (width, height, pixel) Char mapper (int, int)

□ Soblex 역시 classic class 와 같은 방식이지만 mapper가 다른 결과를 나타낸다.

□ Current 라는 integer 변수는 현재 position 을 가르킨다. (artist class 의 getint 멤버 함수 사용) 그리고 x 축 양의 방향으로 인접한 pixel 이 존재한다면 next 라는 integer 타입 변수에 다음 position 을 할당받게 한 뒤, 둘을 비교해서 50 이상 차이가 난다면 | 를 반환하고 그렇지 않다면 빈칸을 반환한다. 만약 현재 위치를 나타내는 x 좌표가 가장 끝을 가리키는 상황에서는 더 이상 비교할 픽셀이 없기 때문에 자동으로 빈칸을 반환하도록 했다.

Sobley (artist 상속)	
private	public
	sobley (int width, int height, cost vector<int>& pixel) : artist (width, height, pixel) Char mapper (int, int)

□ Sobley 는 위와 비슷하지만 x 축 양의 방향으로 인접한 pixel 이 아닌 y 축 양의 방향으로 인접한 pixel 과의 비교를 한 후 50 이 넘으면 | 를 반환하고 그렇지 않거나 맨 마지막 열을 가리키는 상황에서는 빈칸을 반환하도록 했다.

Gradient (artist 상속)	
private	public
	gradient (int width, int height, cost vector<int>& pixel) : artist (width, height, pixel) Char mapper (int, int)

□ Gradient class 에서는 위의 soblex, sobley 뿐만 아니라 x 축 양의 방향으로 인접한 pixel과 y축의 양의 방향으로 인접한 pixel 모두 50 이상 차이가 난다면 + 를 반환한다.

Drawer	
Private	Public
Int width_D Int height_D Vector <char> characters_D	Drawer (artist*) Int getwidth() Int getheight() Vector <char> getchar() Virtual string draw()

□ Drawer 클래스의 private 멤버에는 너비를 할당받는 width_D, 높이를 할당받는 height_D, 그리고 각 픽셀의 문자 값을 저장하는 characters_D 가 있다.

□ Drawer 클래스의 public 멤버에는 constructor, private 멤버에 저장된 값들을 다른 클래스에서 사용할 수 있도록 가져오는 getwidth, getheight, 그리고 getchar 라는 getter 함수가 있다. 마지막으로 drawer 클래스를 상속 받는 클래스 들에서 사용 될 draw 함수가 있다.

□ Drawer 의 constructor 는 artist 포인터 변수를 argument 로 받아온다. Artist class 에 있는 getter 함수들을 이용해 너비, 높이, 문자 벡터를 drawer 클래스의 private 멤버에 저장한다.

□ Drawer 클래스의 draw 함수는 characters_D 에 저장된 문자들을 하나의 string으로 저장하는 역할을 한다. 두개의 for 문과 position integer 타입 변수를 이용해 string type 변수인 result 에 저장한다. Position 변수는 2차원 x, y 값을 이용해 1차원 vector 로 저장되어 있는 문자의 위치를 알 수 있도록 한다. 이는 위에서 사용했던 artist 의 getint 에서 사용한 방식과 같다. 첫번째 행이 끝나고 나면 "₩n" 를 string 에 추가해 나중에 하나의 string 을 프린트 하더라도 줄바꿈이 되도록 했다. 모든 for 문을 실행하고 나면 string 변수를 반환한다. 이렇게 반환된 string 은 메인함수에서의 string output 에 저장된다.

Downsample (drawer 상속)	
Private	Public
	Downsample(artist) String drawer

□ Downsample 은 전체 그림을 ½ 로 줄이는 역할을 한다. 예를 들어 4*4 사진이라면 2*2 로 줄인다. 이때, 홀수 값을 가지는 좌표의 문자들은 무시하고 0을 포함한 짝수 번째의 좌표들만 저장한다.

□ Downsample 의 Constructor 에서는 drawer 의 constructor 를 이용해서 값들을 할당한다.

□ Downsample 의 draw 에서는 drawer class 의 getter 함수들을 이용해 너비, 높이, 각 픽셀 문자를 받아온 뒤 두 개의 for 문을 이용해서 값을 저장한다. 이때, for 문에서 사용되는 i 와 j 문자가 1 씩 증가하는 것이 아니라 2 씩 증가할 수 있도록 한다. 그 외는 위의 drawer 클래스의 draw 와 유사하다.

Upsample (drawer 상속)	
Private	Public
	Upsample (artist*) String draw()

□ Upsample 은 전체 그림을 2배 시킨다. 예를 들어 4*4 그림이라면 8*8 로 늘리는 역할 을 한다. 각 행과 열은 두번씩 반복되어서 저장된다.

□ 위의 class 와 같이 constructor 는 drawer 의 constructor 를 이용한다.

□ Upsample 의 draw 에서는 drawer class 의 getter 함수들을 이용해서 너비, 높이, 각 픽셀 문자를 받아와 저장한다. 그리고 두개의 for 문과 하나의 while 문을 사용해 값을 string 으로 저장한다. 첫번째 for 문에서는 높이를 결정하고 repeat 이라는 변수를 이용해 같은 열을 반복해서 두번 프린트 할 수 있도록 했다. 같은 열 안에서 값이 두번 씩 반복하는 것은 반복문 대신 두번째 for 문 안에서 string 에 두번 추가해주는 방식으로 구현했다.

Scale (drawer 상속)	
Private	Public
Int x_mul Int y_mul Int width Int height Vector <char> characters Void increase_y(string&) Void same_y(string&) Void decrease_y(string &)	Scale (artist*, int, int) String draw

□ 먼저 private 변수에는 각각 늘어날, 혹은 줄어들 scale 을 저장하는 x_mul 과 y_mul 이 있다. 그리고 각 픽셀의 문자는 characters 에 저장될 것이다. Private 멤버 함수들은 draw 를 할 때, y 가 늘어나거나, 같거나, 줄어들 때의 상황에 따라 나눠서 실행하도록 했다.

□ Scale constructor 는 artist* 변수는 drawer constructor 를 이용해서 저장하고 두개의 int 변수인 x, y 축소/확대 값은 각각 x_mul, y_mul 에 저장한다. 그리고 drawer 의

getter 함수를 이용해서 scale 의 private 멤버 변수인 width, height, characters 에 저장했다. 이는 scale 클래스의 다른 여러 함수들에서도 쉽게 접근하기 위해서 이번 클래스의 private 멤버로 저장했다.

- Scale draw 는 if 문을 이용해 y 축으로 확장되는 상황, y 축이 확장되지도 않고 축소되지도 않는 상황, 그리고 y 축으로 축소되는 상황을 고려해 각각의 private 멤버 함수를 실행한다.

- 먼저 increase_y, 즉 y 축으로 확장해야 하는 상황에서는 for 문 두개를 이용해 같은 열을 y_mul 만큼 반복해서 프린트 하도록 하고 두 개의 for 문 안에서 x축으로 확대되는 상황, 축소되는 상황, 그리고 그대로 유지 되는 상황에 따라 각각 이중 for 문으로 반복해서 프린트 하거나 하나의 for 문에서 x_mul 의 절댓값으로 증가하거나 또는 한번씩 프린트 하도록 한다.

- Decrease_y 와 same_y 도 위와 같이 각각 행을 몇번 프린트 할 것인지 정한 뒤 if 문을 이용해서 3가지 x_mul 에 따른 상황을 각각 고려해주었다.

- 각 행이 끝날 때 마다 "₩n" 을 추가해 2차원으로 출력될 수 있도록 했다.

- 이렇게 draw 는 메인에서 d->draw() 를 수행했을 때 실행되고 p.write_result() 에서 output 이 저장될 txt 파일에 문자가 프린트 될 것이다.

3. 토론 및 개선

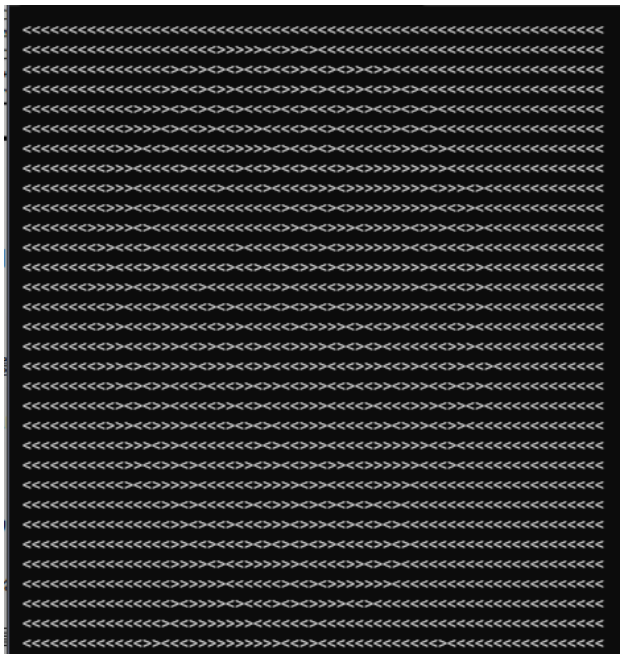
- Drawer 의 생성자가 artist 가 아닌 artist* (포인터)로 인자를 받는 이유는 데이터 손실을 막기 위해서 이다. 클래스 간의 객체를 인자로 전달할 때, 전달되는 객체의 크기가 함수 매개변수로 할당된 메모리의 양보다 큰 경우에 데이터 손실이 일어날 수 있다. 또한, 값으로만 전달하는 경우 데이터 손실이 일어나지 않더라도 함수는 객체의 복사본을 받는 것이기 때문에 객체가 큰 메모리를 사용한다면 이는 비효율적일 것이다.

Mine (artist 상속)	
Private	Public
	Mine (int, int, const vector <int>&) Mapper

- 자신만의 style을 만들기 위해서 artist class 를 상속 받는 클래스 mine 을 만들었다. 그리고 메인 함수에서 if 문을 이용해 적절한 config 에 맞는 artist 객체인 style을 동적할당 하기 위해서 else if 문에서 mine 을 추가해줬다.

- 추가된 코드: oop assn 3-1.cpp 에서 line 46~48, 432~446, 그리고 artist.hpp 에서 line 112~129 가 추가되었다.

- Sobelx 에서 처럼 현재 좌표 값을 저장하는 position 변수와 x축 양의 방향으로 인접한 픽셀 값을 할당 받는 next 변수 값을 이용해 만약 현재 좌표 값이 크면 ">" 를 저장하고 인접한 값이 더 크면 "<" 를 저장하는 함수를 만들었다. 밑의 두 사진은 각각 터미널과 output.txt 에 저장된 그림을 보여준다.



- 이번 assignment 에서는 main 함수를 수정할 수 없기 때문에 artist 가 construct 될 때 바로 integer vector 를 char vector 로 바꿔주었다. 하지만 이 과정을 draw 에서 한다면 scale 을 실행할 때 4중 for 문을 이용하지 않고 더 간략하게 할 수 있을 것 같다.

- 그리고 scale 함수를 구현할 때 따로 artist 에서 getter 함수를 이용해 다시 값을 저장해주었는데 그렇게 되면 character 를 저장하는 vector 역시 전부 copy 해야 한다. 메모리 낭비를 줄이기 위해서 A->getwidth() 등을 이용하거나 상속관계 일 때 접근이 가능하게 하는 protected 를 사용하면 좋을 것 같다.

- Destructor 를 구현하는 데에 어려움이 있었다. 위에서 언급한 것 처럼 draw class 에서 artist 를 저장한 것이 아니라, draw 의 destructor 가 call 되었을 때, artist 를 할당해제 시킬 수 없었다. 그러므로 draw 가 construct 될 때, artist 에서 필요한 정보들이 복사 되기 때문에, constructor 에서 delete (artist name) 을 해줬다. 하지만 delete 는 할당 해제 함수이기 때문에 destructor 에 들어가는 것이 더 적절할 것 같다.

4. 참고 문헌

- Stoi 함수 : https://cplusplus.com/reference/string/stoi/

- Push_back 함수: https://cplusplus.com/reference/vector/vector/push_back/
- Append 함수 : https://cplusplus.com/reference/string/string/append/
- Empty 함수: https://cplusplus.com/reference/string/string/empty/
-