

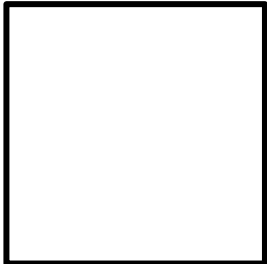
컴퓨터 시스템 2020 기말 프로젝트-1

■ 주제: MASM을 활용하여 간단한 그래픽 작업을 수행하는 프로그램 만들기

내용: 프로젝트의 목표는 그림판 (mspaint)의 '색 칠하기 (채우기/페인트 통 붓기)' 기능을 어셈블리 프로그램으로 구현하는 것이다. 먼저 N x M 바둑판 형태로 주어진 이미지 포맷 데이터를 입력으로 받아, 사용자가 지정한 지점/위치에 대해 그 지점과 연결된 같은 색상 (데이터 값) 영역을 특정 색상으로 칠하는 기능을 수행하여, 그 결과를 새로운 이미지 파일로 출력하는 간단한 프로그램을 설계함으로써 어셈블리 프로그래밍을 통한 기본 입출력, 파일 입출력, 프로시저 호출, 자료형 활용 등 한 학기 동안 배운 지식을 활용하는 것이 본 프로젝트의 목표이다.

■ 구현 사항:

- 1) 프로젝트의 주요 단계는, ① 주어진 이미지 파일을 읽어 드린 후, ② 사용자가 명시한 특정 지점으로 이동하여, ③ 해당 위치와 같은 값의 연결된 영역을 특정 색상으로 채우는 동작을 수행하고, ④ 그 결과를 이미지 파일로 출력하는 프로그램을 구현하는 것이다.
- 2) 이미지 파일 포맷 [입 출력 공통] (①, ④)
 - a) 화일의 첫 라인에는 N-width, M-height에 대한 정보가 다음과 같이 표시된다 (50x50 image 예시).
 - i) 예시) 50,50

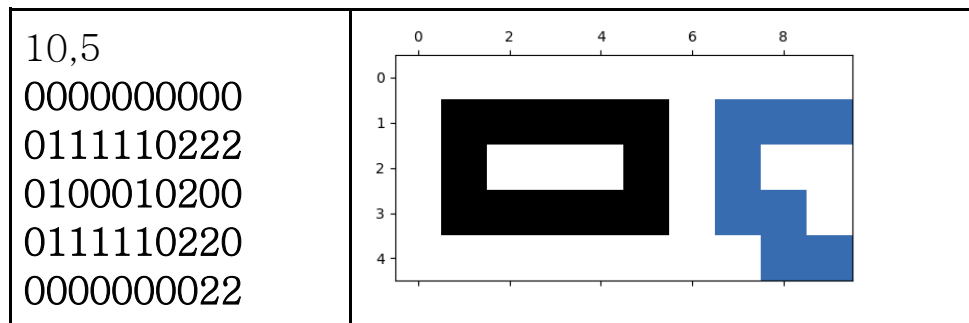
N-width	
	line
	123450
	1 50,50
	2 0000 ... 0000
	3 0111 ... 1110

	51 0000 ... 0000
M-height	

- b) 그 다음 라인 (2nd line ~)부터는 M줄당 N개의 색상 정보값 (0~6)에 해당되는 정수가 입력된다.
 - i) 예시) 10x5 크기의 이미지 데이터와 인접/연결된 색상으로 표시한 결과 예시

<sample.txt>

<python view_img.py sample.txt 실행 결과>



- c) 해당 프로젝트에서 구현하는 프로그램의 입력/출력은 모두 상기 포맷에 맞추어 수행해야 한다.
- d) 주의: 최상단, 최좌측의 좌표는 (0, 0)이며, 최하단, 최우측의 좌표는 (N-1, M-1)이다.

- 3) 커맨드 라인 상에서의 입력 조건은 다음과 같다 [프로그램이 위치한 Directory에서 실행] (②, ③):

a) ./mypaint [source img file] [output img file] [x] [y] [color]

- i) [source img file]: 현재 디렉토리에 저장된 수정하고자 하는 이미지 파일의 이름을 입력 받는다.
- ii) [output img file]: 현 디렉토리내 결과로 출력하고자 하는 이미지 파일명을 입력 받는다.

- iii) [x], [y]: 색을 칠하고자 하는 영역에 대한 위치 정보 좌표 (x, y)를 입력 받는다.
- iv) [color]: 칠하고자 하는 색상에 대한 정보를 입력 받는다.
- v) 상기 입력 조건은 반드시 준수되어야 한다.
- vi) 생성된 바이너리의 이름은 **mypaint**로 한다.

4) 색상 채우기 동작을 위한 필수 구현 사항 (③):

- a) 이미지 파일은 그리드 형태로 픽셀들이 배열되었다는 가정하에, 색상이 '칠해진다 (fill)'라는 동작은 다음과 같이 정의된다:
 - i) 입력 받은 [x][y]를 중심으로, 해당 지점[x,y]과 (1) 사방으로 인접한 픽셀들에 한해서, 그리고 (2) 지점 [x, y]와 동일한 색상값을 가진 픽셀들을 '같은 영역'이라 정의한다. 또한 '같은 영역'에 속한 모든 픽셀들에 대해서도 마찬가지로 (1) 사방으로 인접하였으며, (2) 동일한 색상값을 가진 픽셀들 또한 '같은 영역'에 재귀적으로 속하게 된다.
 - mspaint (윈도우 그림판) 동작 참고: <https://youtu.be/6rRslb1eR-Y?t=147>
 - 쉽게 표현하자면, 다른 색의 픽셀들로 둘러싸였으면서 같은 색으로 구성된 부분이 같은 영역이다.
 - ii) (1) 다음과 같은 5 x 5 데이터 예시에서 사방으로 인접한 경우는 다음과 같다.
 첫 줄의 위치 좌표는 (0,0), (1,0), (2,0), (3,0), (4,0)으로 표시된다.
 지점 **P(2,2)**을 기준으로 '사방으로 인접한' 픽셀은 **O**로 표기된 (2,1), (2,3), (1,2), (3,2)이다.

	X	O	X	
	O	P	O	
	X	O	X	

iii) '같은 영역'의 예시는 아래 그림과 이에 대한 설명과 같다:

- case 1) (0,0)을 기준으로 같은 영역은 (1) 사방으로 인접하였으며, (2) 같은 색상인 '1'을 가진 붉은색으로 표시된 1번 픽셀 영역이다.
- case 2) (3,1)을 기준으로 같은 영역은 (1) 사방으로 인접하였으며, (2) 같은 색상인 '1'을 가진 검은색으로 표시된 1번 픽셀 영역 뿐이다.
- case 3) 그 이외의 영역들의 경우, case 1, 2와 같이 같은 색상을 가졌음에도 사방으로 인접하지 않아 분리가 되는 영역의 경우가 발생하지 않아, 모두 1색 1영역을 가진 경우이다.
 (따라서 (0,0)에 새로운 색을 칠하더라도 (3,1)은 바뀌지 않는다. 당연히, (3,0)에 새로운 색을 칠하더라도 (3,1)은 색상이 바뀌지 않는다.)

1	1	1	3	3
1	1	2	1	3
1	1	2	3	3
2	2	2	3	3
2	2	2	3	3

- iv) 즉, '색상이 칠해진다' 라는 동작은 입력 받은 특정 지점을 중심으로 '같은 영역'에 해당되는 모든 픽셀들을 사용자가 지정한 새로운 색상으로 덧씌우는 것을 의미한다.

5) 구현 방식

- a) 색 채우기 기능에 대한 구현 방식 (활용 알고리즘, 방법론)에는 제약이 없음.
 - i) 단, **MASM Directives (.IF, .REPEAT, .WHILE, .FOR 등) 사용 불가**
- b) Input file의 크기는 50 x 50으로 제한 (고정) 함. 하단부에 예시로 주어진 동작들을 합리적인 시간 (< 3 second) 안에 수행하지 못할 경우 감점 대상임.
- c) 프로그램의 주요 동작 파트 별로 간단한 주석 작성 (주요 수행 동작, 사용 변수에 대한 명시 등)

6) 색 종류: 8가지 (0~7) (③)

- 0: 흰색, 1: 검정색, 2: 청색, 3: 녹색, 4: 황색, 5: 적색, 6: 주황색, 7: 보라색

■ 제출물

- 1) 과제 보고서: 프로그램 수행에 대한 논리적 흐름 서술 (입/출력, 핵심 동작 파트) 및 고안한 색 채우기 기법에 대한 방법론을 코드와 같이 설명
- 2) 프로그램 파일 (테스트 입력으로 검증 예정)
- 3) 구동 결과 화면 캡처

■ 제출기한

- 1) 2020년 6월 15일 17:00시 까지

■ 예시: 캐릭터 색칠하기

- 원본 화일 before.txt (size: 30x30)에 다음과 같은 과정을 통해 색 채우기를 한 결과가 after.txt이다. (첫번째 작업에서 산출된 after.txt를 계속 덮어쓰기를 하며, 프로그램을 총 3번 반복 실행한 결과이다.)
- 1) (0, 0) 지점에 3 (녹색)으로 색 칠하기 → ./mypaint before.txt after.txt 0 0 3
- 2) (10, 5) 지점에 2 (청색)으로 색 칠하기 → ./mypaint after.txt after.txt 10 5 2
- 3) (20, 12) 지점에 6 (주황색)으로 색 칠하기 → ./mypaint after.txt after.txt 20 12 6



<before.txt>



(1)



(2)



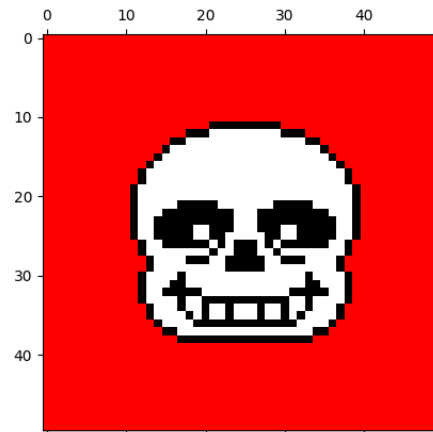
(3)-<after.txt>

■ 참고사항:

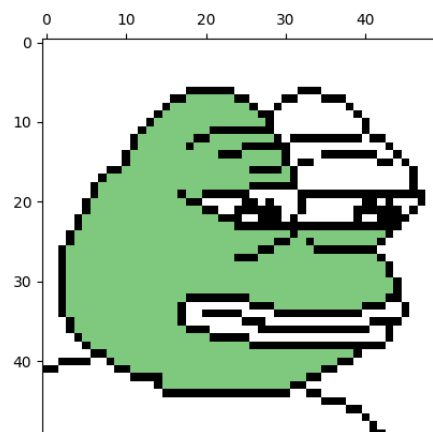
- 1) 과제 수행의 편의를 위하여 해당 이미지 데이터 파일을 시각화하기 위한 파이썬 프로그램 (view_img.py) 및 예시의 이미지 파일 (before.txt)이 제공된다.
 - a) 의존성: matplotlib / numpy package
 - b) 채점의 경우 별도의 채점용 이미지 파일(50x50)을 통해 이루어진다.
 - c) 과제 명세서 상의 예시들의 경우, 이해의 편의를 위하여 소규모 이미지를 활용하였으나, 실제 채점은 50x50 고정 이미지로 이루어지며, 제출시에도 이 포맷을 준수하여야 한다.
- 2) view_img.py의 실행 방법은 다음과 같다 (img_file에는 타겟 이미지 파일의 path를 입력).
 - a) python view_img.py *img_file*
 - b) 실행시 상단 예시와 같이, before.txt 및 after.txt와 같은 화면이 출력된다.
- 3) 참고자료:
 - a) <https://lodev.org/cgtutor/floodfill.html>
 - b) <https://www.geeksforgeeks.org/flood-fill-algorithm-implement-fill-paint/>
- 4) 참고 입력 파일:
 - a) 예시1

sans.txt

python view_img.py sans.txt

[illegible]

b) 예시2

[illegible]

[illegible]