시뮬레이션 기초 및 실습 HW 3

E-learning으로 1과2를 압축하여 최종적으로 하나의 파일로 제출 (예: 학번_이름.zip)
1. 각 문제 (예: 1은 Problem_1.m로 저장)에 대한 소스 코드 제출. 간단한 주석 포함
2. 풀이과정을 담은 리포트. PDF 형식 (.pdf). 리포트에는 본인이 작성한 알고리즘에 대한 설명, 실험 결과에 대한 해석 및 토의가 반드시 포함되어 있어야 한다. 설명 및 토의 과정 없이 코드만 적으면 0점

1. Matlab에는 '^'을 통하여 x의 n승을 계산할 수 있다. Matlab '^'을 사용할 수 없다고 가정 하자. 재귀 함수로 x의 n승을 계산하는 방법이 있다. 아래 예는 x^{21}을 재귀로 계산한 예이다

$$x^{21} = (x^{10})^2 \cdot x$$
$$\rightarrow x^{10} = (x^5)^2$$
$$\rightarrow x^5 = (x^2)^2 \cdot x$$
$$\rightarrow x^2 = (x)^2$$

이 함수의 재귀적 정의는 아래와 같이 정의되어 있다.

$$f(x,n) = \begin{cases} 1 & ,if\ n = 0 \\ f(x,\frac{n}{2}) \cdot f(x,\frac{n}{2}) & ,\text{if } n > 0 \text{ and } n\text{이 짝수} \\ f(x,\frac{n-1}{2}) \cdot f(x,\frac{n-1}{2}) \cdot x & ,\text{if } n > 0 \text{ and } n\text{이 홀수} \end{cases}$$

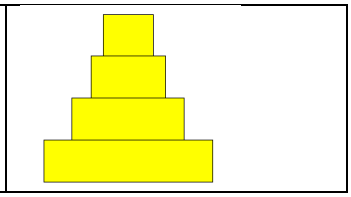
이를 이용하여 아래와 같은 재귀 함수 PowerP(x, n)을 작성해 보자. 어떻게 알고리즘을 작성했는지 설명하자. 작성한 코드가 잘 동작하는지 확인하기 위하여 PowerP(3, 4)를 실행한 출력 결과 (Matlab 명령창)를 캡쳐하여 보여주자.

```
function y=PowerP(x, n)
% y=x^n, 여기서 n은 0보다 크거나 같은 정수
```

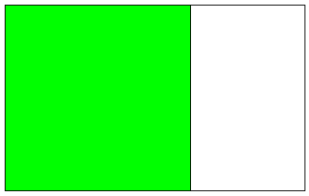
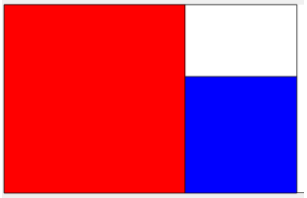
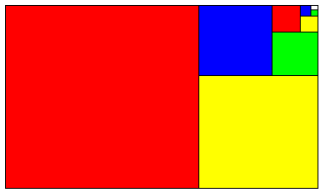
2. 아래 왼쪽에 정의된 'DrawRect.m' 함수는 가로 길이 L, 세로 길이 W인 직사각형을 색 c를 이용하여 그리는 함수이다. 이 직사각형의 왼쪽 아래 코너 점의 (x, y)좌표는 (a, b)이다. 예를 들어 DrawRect(2, 1, 2, 2, 'y')는 직사각형의 왼쪽 코너 아래 점이 (x, y)=(2,1)이고 가로 길이=2, 세로 길이=2이며 노란색 직사각형을 그린다.

'DrawRect' 함수를 반복 호출하고 while 문을 이용하여 아래 오른쪽과 같은 피라미드 형태를 그리고자 한다. 이 피라미드의 제일 아래 직사각형의 왼쪽 아래 코너 좌표는 (x, y)=(0, 0)이다. 이 피라미드의 제일 아래 직사각형은 가로 길이가 W이고 세로 길이가 H이다. 이 두 W와 H 값들은 'input' 함수를 이용하여 사용자에게 직접 입력 받는 값이다. 단, 사용자가 입력 시 H값은 W값보다 항상 같거나 작은 값이라고 가정하자. 각 피라미드의 직사각형의 높이는 모두 H이고. 피라미드의 제일 아래로부터 한 칸 식 위로 올라갈수록 직사각형의 가로 길이는 바로 아래 직사각형의 가로 길이의 2/3 이다. 단, 피라미드의 가장 위의 직사각형의 가로 길이는 반드시 H보다 작도록 피라미드를 만들고자 한다. 어떻게 알고리즘을 작성했는지 설명하자. L=20, H=1인 경우에 대하여 본인이 작성한 코드를 사용하여 피라미드 형태를 만든 출력 결과를 캡쳐하여 보여주자. 실행 시 총 몇 개의 직사각형이 보이는가?

```
function DrawRect(a, b, L, W, c)
x = [a a+L a+L a   ];
y = [b b    b+W b+W];
fill(x,y,c)
```



3. 어떤 직사각형에 subdivision 과정을 한번 수행하면 아래 왼쪽 그림처럼 정사각형과 직사각형으로 나뉘어질 수 있다. 이 subdivision 과정을 한번 더 수행하면 남은 직사각형 부분을 아래 가운데 그림처럼 다시 정사각형과 직사각형으로 나눌 수 있다. 이를 계속 반복하여 남은 직사각형 부분을 정사각형과 직사각형으로 나눌 수 있다.

이를 수행하는 재귀 함수 'PartitionRect.m' 을 divide-and-conquer 방법으로 작성하고자 한다. 이 함수의 원형은 아래 왼쪽과 같이 정의되어 있다. 이 함수는 위에서 설명한 직사각형의 subdivision 과정을 L번 수행하는 함수로 직사각형의 왼쪽 아래 코너 점의 좌표는 (a, b)이고 직사각형의 base (밑변, 가로)와 height (높이, 세로)가 입력인 함수이다. 위의 왼쪽 그림은 L=1, 위의 가운데 그림은 L=2, 위 오른쪽 그림은 L=8인 경우의 실행 예이다. subdivision 수행 시 생성되는 정사각형 부분의 색은 무작위로 선택하는데 'r', 'g', 'b', 'y'가 같은 확률로 선택하도록 하고 subdivision 수행시 남은 직사각형 부분은 흰색 ('w')이 되도록 하자. 실험 시 직사각형의 base=1.6180, height=1.0, a=0, b=0으로 설정하자. 아래 오른쪽에 있는 main 함수를 실행한 결과를 출력해보고 실행 결과 (그림: 총 5개)를 레포트에 출력하자. 어떻게 알고리즘을 작성했는지 리포트에 설명해보자.

Hint: divide and conquer 방법으로 코드를 작성시 한가지 방법은 재귀 단계를 작성시 base가 height보다 큰 경우, base가 height보다 작거나 같은 경우를 모두 생각해 보는 것이다.

PartitionRect.m	main.m

function PartitionRect(a, b, base, height, L)
if L==0 % 기본 단계
?
else % 재귀 단계
?
end
	clear;
	close all;
	a=0;
	b=0;
	base=1.6180;
	height=1;
	for L=1:5
	figure;
	hold on;
	PartitionRect(a,b,base,height,L)
	end