

1.

컴퓨팅 사고의 구성요소는 분해, 추상화, 패턴인식, 알고리즘 이렇게 총 4가지로 구성이 되어 있다.

- 분해는 주어진 문제를 해결하기 위해 쉬운 작은 단위의 문제로 나누는 것을 말하며, 예를 들어서 라면을 만들고자 할 때, 스프 넣기, 면 넣기, 추가 재료 넣기 등 작은 것에서(쉬운 것) 먼저 해겨하고 난 뒤 나머지 어려운 부분으로 단계적으로 해결하는 것을 말한다.

- 추상화는 문제에서 중요하지 않은 부분을 제거하고 난 뒤에 중요한 특징만 남겨 단순화 시키는 것으로 흔히 우리가 알고 있는 픽토그램이나 지하철 노선도 같이 본연이 가지고 있는 특징만을 그대로 보여주는 것을 말한다.

- 패턴인식은 주어진 데이터를 특징별로 나누어 의미있는 패턴이 있는 지 찾는 것으로 tv 속의 rgb 값이 어떠한 패턴을 통하여 유저들이 원하는 정보를 출력하는 것을 말한다.

- 알고리즘은 문제해결을 위해 일련의 절차나 방법을 공식화한 형태로 표현한 것으로 플로우 차트와 같은 순서도로 단계적으로 문제를 해결하는 것을 뜻한다.

2.

컴퓨팅 사고가 필요한 이유는 문제 해결 과정에서 우위를 점할 수 있기 때문입니다.

즉, 컴퓨팅 사고력은 인간이 사용하기는 하지만 인지하지 못했던 사고 과정에 필요한 순차, 반복 및 알고리즘 구현 방법 등과 같은 기법들을 체계적으로 학습할 수 있습니다. 그리고 이와 같이 다양한 기법들을 사용하여 문제 해결에 보다 체계적인 접근이 가능하며 빠르게 해결이 가능하다는 것입니다.

최근에 융합 및 복합 이라는 단어가 광범위하게 사용되고 있는데, 서로 다른 전공과 교과목들에게 접목이 가능하다는 것도 한 몫을 합니다.

융.복합형 인재양성이 어느 때보다 중요해진 현시점에서 컴퓨팅 사고력은 실생활에서부터 다양한 학문 분야까지, 단순한 문제에서 복잡한 문제까지 직면한 모든 문제들을 해결하기 위해 필요한 역량이 되었기에 필수적인 역량이 되었습니다.

3.

- `t.shapesize(w,h,b)` : 거북이(펜)의 크기를 변경함
- `t.backward(angle)` : 거북이가 향하는 반대방향으로 거북이를 지정한 거리 만큼 뒤로 이동시킴
- `t.setheading(to_angle)` : 거북이의 방향을 `to_angle`로 바라보는 방향으로 설정함
- `T.circle(number)` : 주어진 반지름으로 원을 그림
- `t.clear()` : 거북이 그림을 화면에서 삭제함
- `t.setposition(x,y)` : 거북이를 절대 위치(`x,y`)로 보냄, 펜이 내려져 있으면 선을 그림

4

- import keyword
- keyword.kwlist

5.

- 초기값 $x = 1000$
- 초기값 $x = 1002$
- 초기값 $x = 1004$

6. 소스 코드 및 결과화면

```
kor = [] # 국어 점수를 담을 공백 리스트
en = [] # 영어 점수를 담을 공백 리스트
py = [] # 파이썬 점수를 담을 공백 리스트
student = 5 # 학생 수
Psum = 0 # 학생 5명의 파이썬 점수 합
Ksum = 0 # 학생 5명의 국어 점수 합
Esum = 0 # 학생 5명의 영어 점수 합

for i in range(student):
    na = int(input(str(i+1) + "번 학생의 국어 성적: "))
    nb = int(input(str(i+1) + "번 학생의 영어 성적: "))
    nc = int(input(str(i+1) + "번 학생의 파이썬 성적: "))

    kor.append(na) # 입력 받은 학생들의 국어 점수 값을 넣음
    en.append(nb) # 입력 받은 학생들의 영어 점수 값을 넣음
    py.append(nc) # 입력 받은 학생들의 파이썬 점수 값을 넣음
    print("-----")

print("-----")
print("번호\t국어\t영어\t파이썬")
print("-----")

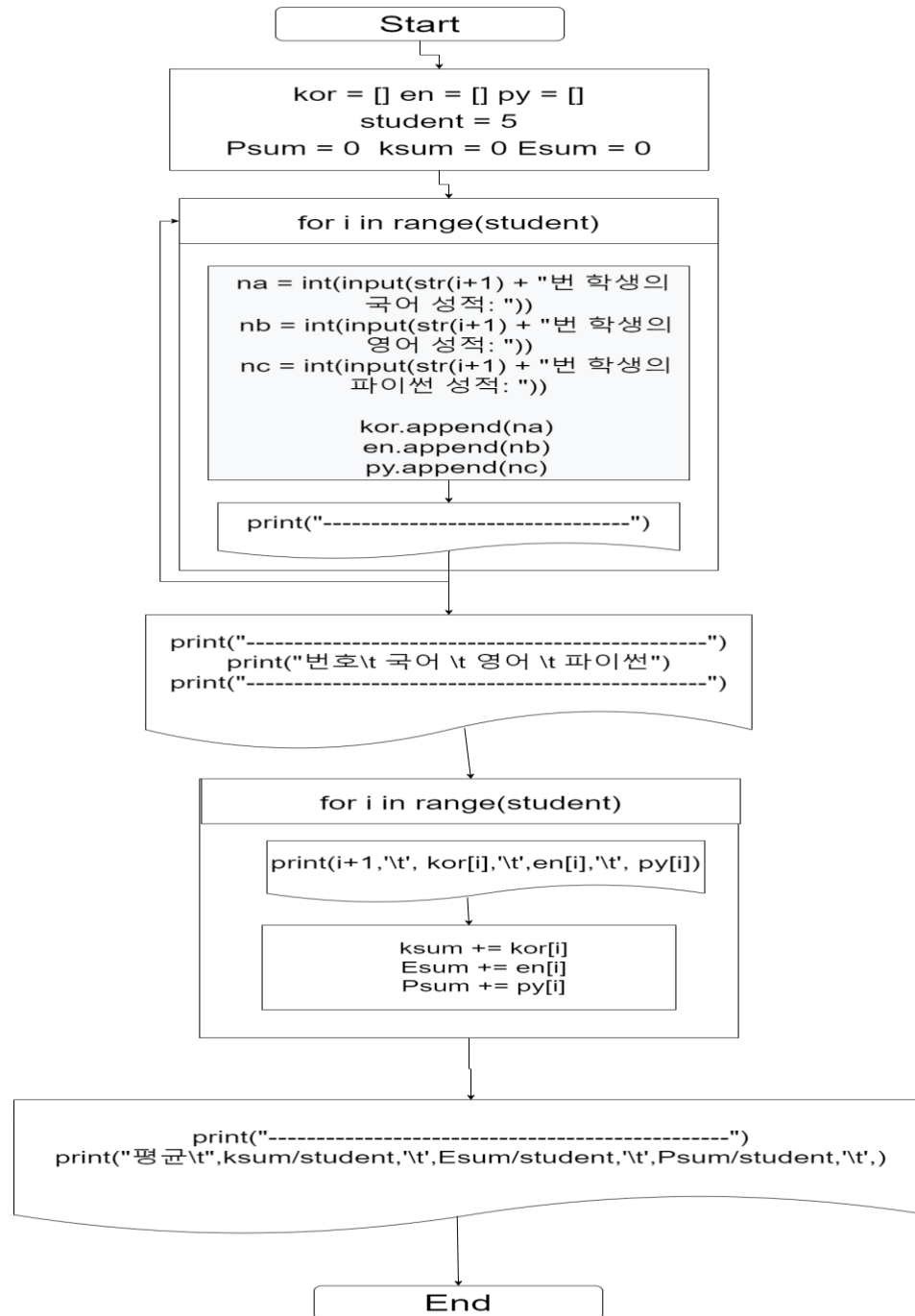
for i in range(student): # 학생 수 만큼 반복을 함
    print(i+1, '\t', kor[i], '\t', en[i], '\t', py[i]) # 학생 번호, 국어 점수, 영어 점수, 파이썬 점수 출력
    Ksum += kor[i] # 학생들의 국어 점수 합
    Esum += en[i] # 학생들의 영어 점수 합
    Psum += py[i] # 학생들의 파이썬 점수 합

print("-----")
print("평균\t", Ksum/student, '\t', Esum/student, '\t', Psum/student, '\t')
# 학생들의 국어 점수 평균, 영어 점수 평균, 파이썬 점수 평균 tab을 통하여 각각 출력
```

6번 결과화면

```
Python 3.8.0 Shell
File Edit Shell Debug Options Window Help
===== RESTART: C:\Users\booti\Desktop\문제 6번.py =====
1번 학생의 국어 성적: 78
1번 학생의 영어 성적: 6
1번 학생의 파이썬 성적: 10
-----
2번 학생의 국어 성적: 79
2번 학생의 영어 성적: 34
2번 학생의 파이썬 성적: 66
-----
3번 학생의 국어 성적: 55
3번 학생의 영어 성적: 23
3번 학생의 파이썬 성적: 19
-----
4번 학생의 국어 성적: 55
4번 학생의 영어 성적: 65
4번 학생의 파이썬 성적: 79
-----
5번 학생의 국어 성적: 100
5번 학생의 영어 성적: 95
5번 학생의 파이썬 성적: 95
-----
번호      국어      영어      파이썬
-----
1          78        6         10
2          79        34        66
3          55        23        19
4          55        65        79
5          100       95        95
-----
평균      73.4      44.6      53.8
>>>|
```

6. 알고리즘

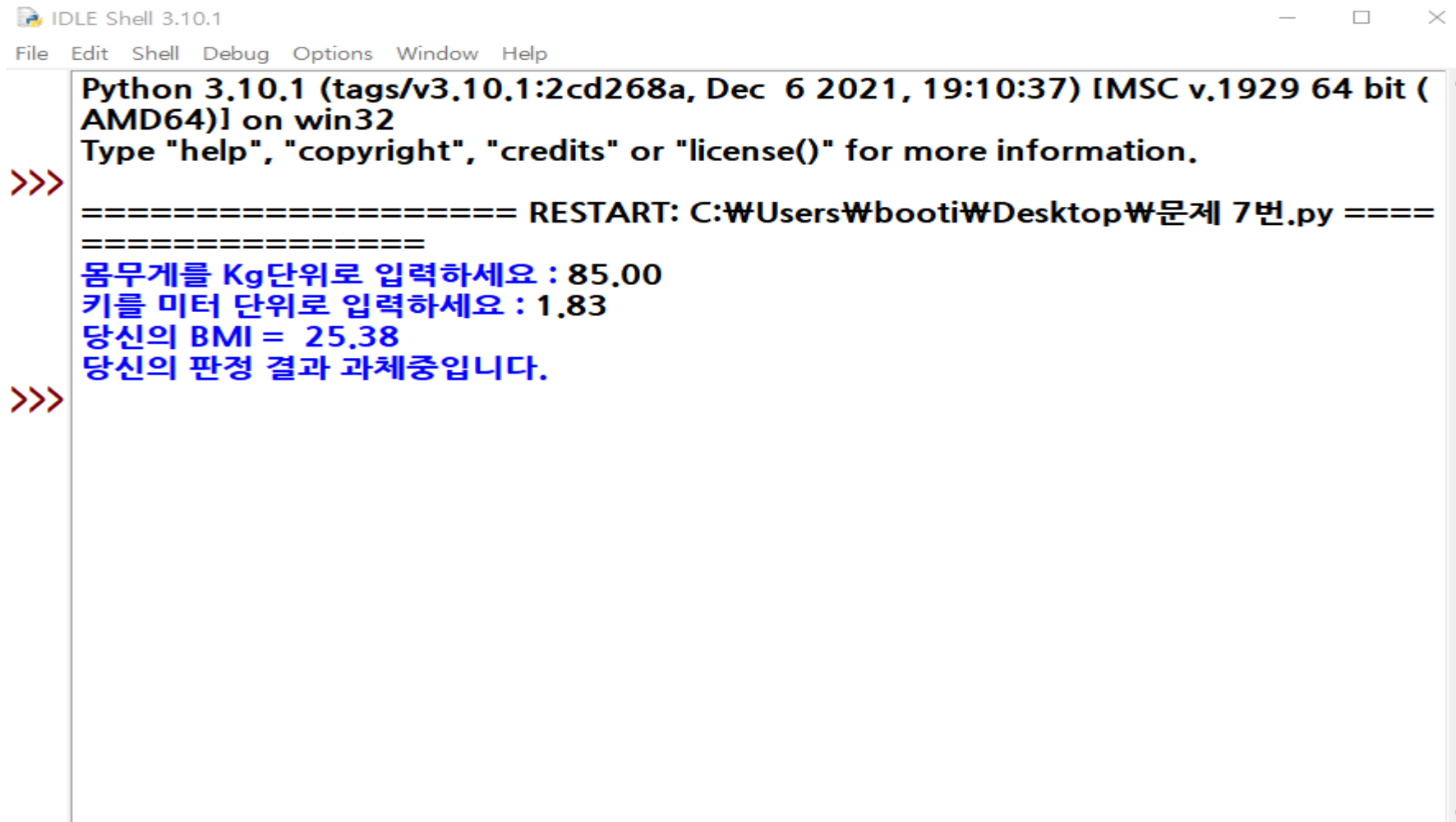


7. 소스 코드 및 결과화면

```
weight = float(input("몸무게를 Kg단위로 입력하세요 : ")) # 입력받을 몸무게 값, 실수형으로 선언
height = float(input("키를 미터 단위로 입력하세요 : ")) # 입력받을 키 값, 실수형으로 선언
bmi = (weight / (height**2)) # bmi 계산
print("당신의 BMI = ", round(bmi, 2)) # 현재 자신의 bmi 출력, 소수 둘째자리까지 반환

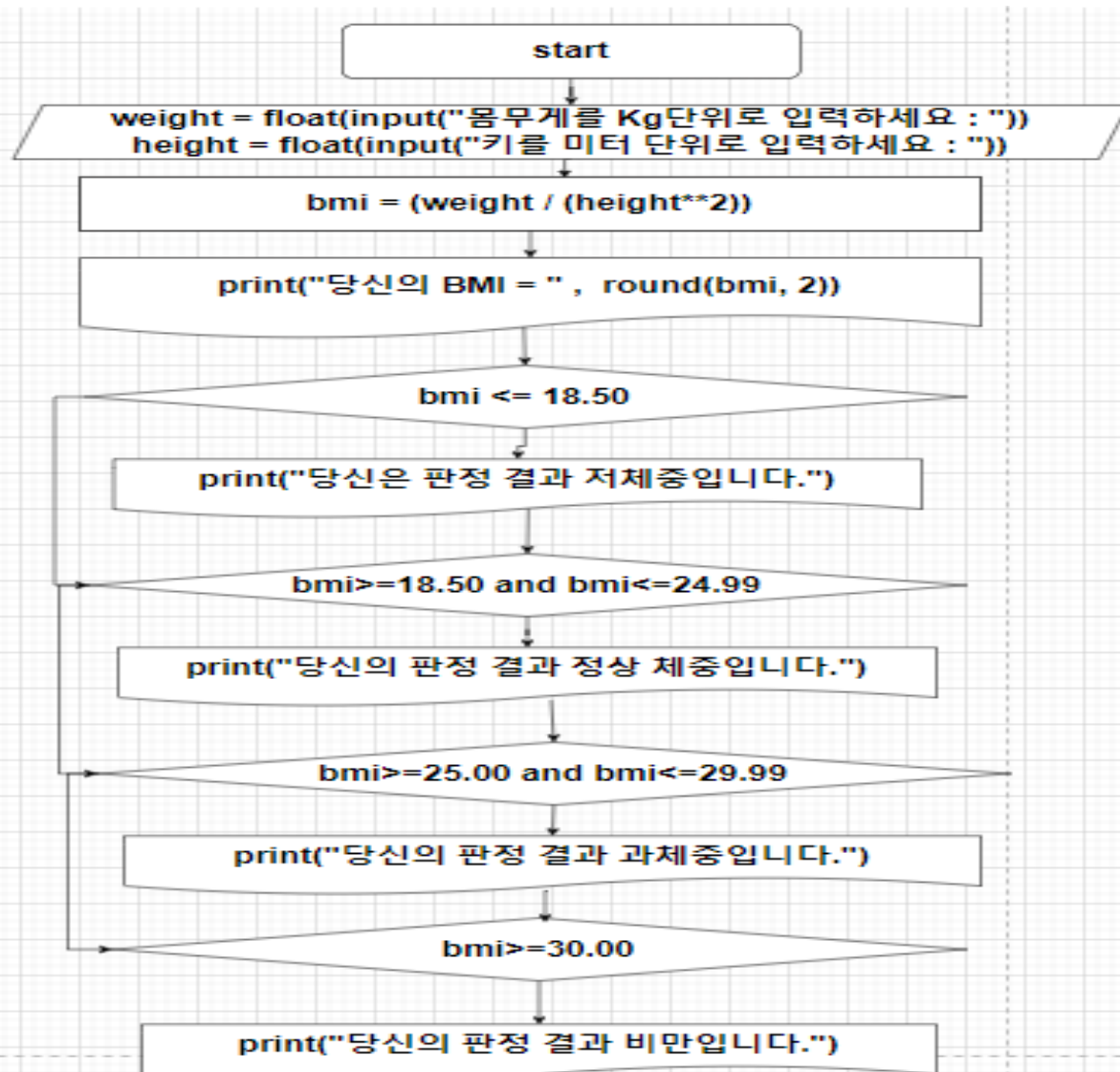
if bmi <= 18.50: # 18.50 이하이면
    print("당신은 판정 결과 저체중입니다.") # 저체중
elif bmi >= 18.50 and bmi <= 24.99: # 18.50 이상 24.99 이하이면
    print("당신의 판정 결과 정상 체중입니다.") # 정상체중
elif bmi >= 25.00 and bmi <= 29.99: # 25.00 이상 29.99 이하이면
    print("당신의 판정 결과 과체중입니다.") # 과체중
elif bmi >= 30.00: # 30.00 이상이면
    print("당신의 판정 결과 비만입니다.") # 비만
```

7. 소스 코드 및 결과화면



```
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Wbooti\Desktop\문제 7번.py =====
=====
몸무게를 Kg단위로 입력하세요 : 85.00
키를 미터 단위로 입력하세요 : 1.83
당신의 BMI = 25.38
당신의 판정 결과 과체중입니다.
>>>
```

7-1. 알고리즘



8. 소스 코드 및 결과화면

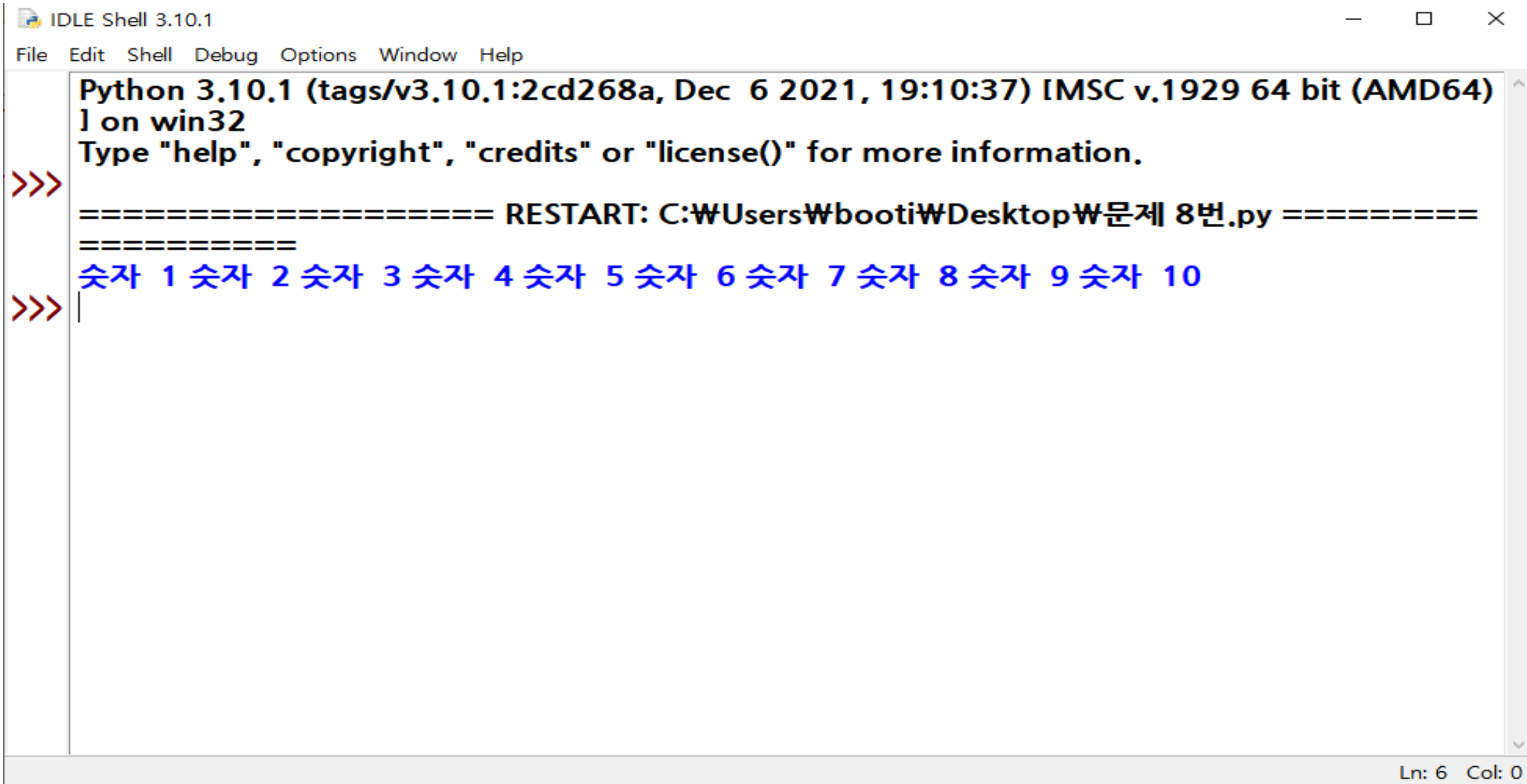
문제 8번.py - C:\Users\booti\Desktop\문제 8번.py (3.10.1)

File Edit Format Run Options Window Help

```
from time import sleep # time sleep 모듈 불러옴
i = 1 # 반복 변수 초기화 --> 1
while i<=10: # 10보다 작거나 같을 때까지 반복
    print("숫자 ", str(i), end=' ') # 숫자 1 숫자 2 숫자 3 ... 숫자 10 (end = ' ' --> 한 줄로 출력하도록 함)
    sleep(1) # 1초 동안 대기
    i+=1 # 값 1씩 증가
```

Ln: 6 Col: 18

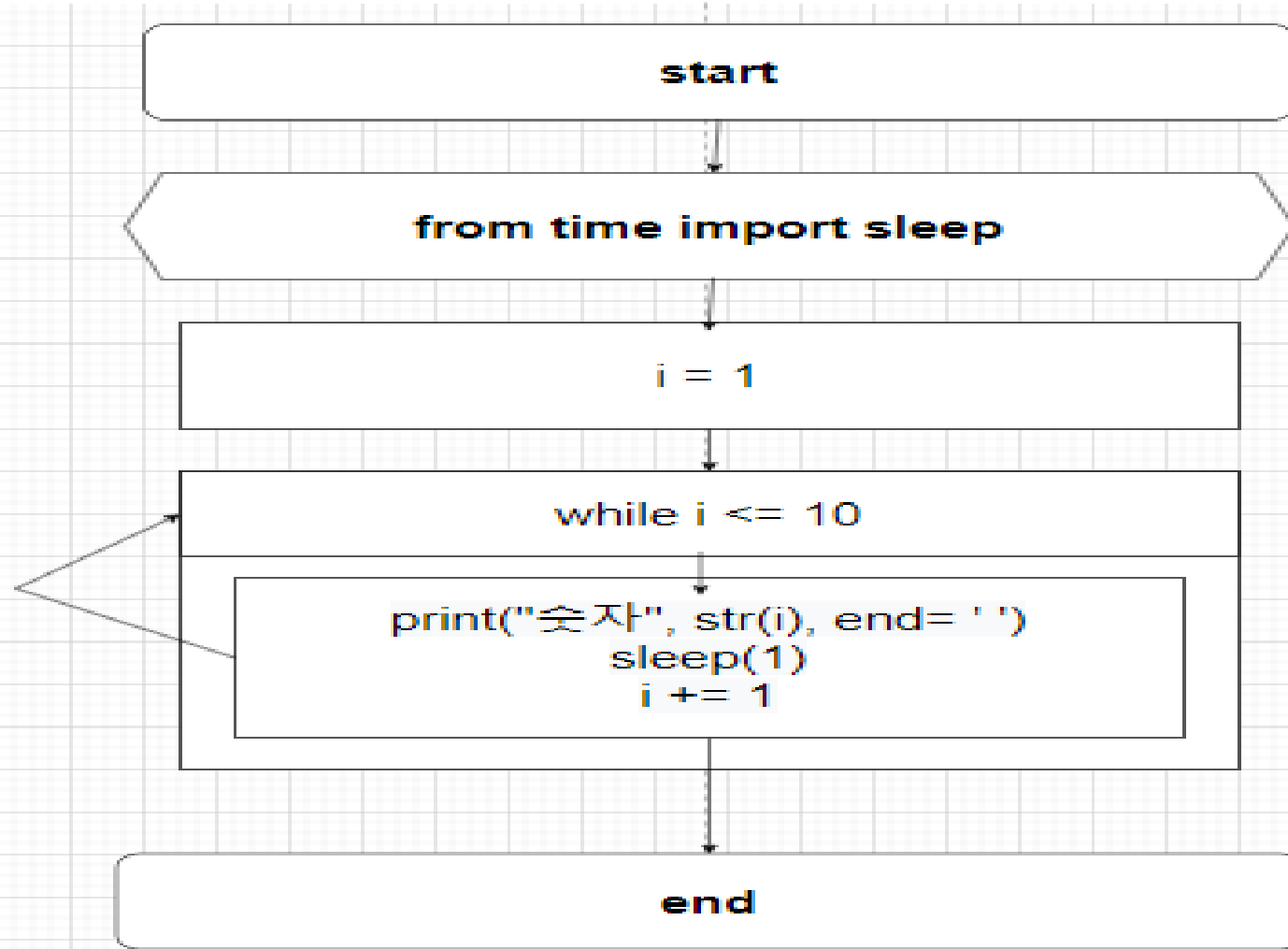
8. 소스 코드 및 결과화면



```
IDLE Shell 3.10.1
File Edit Shell Debug Options Window Help
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Wbooti\Desktop\문제 8번.py =====
>>> 숫자 1 숫자 2 숫자 3 숫자 4 숫자 5 숫자 6 숫자 7 숫자 8 숫자 9 숫자 10
>>> |
```

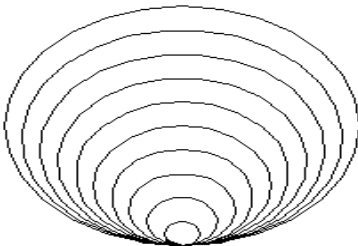
Ln: 6 Col: 0

8-1. 알고리즘

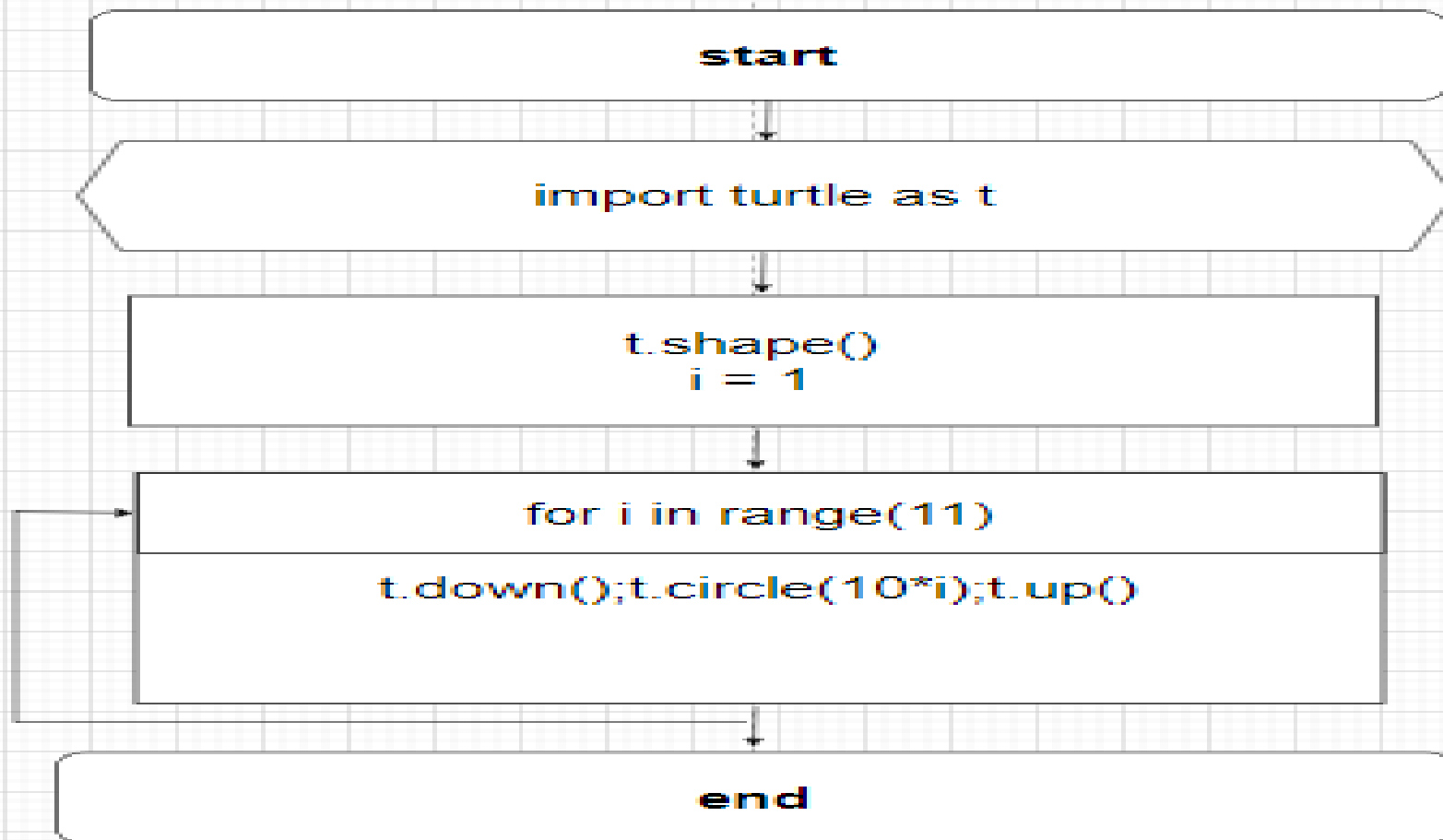


9. 소스 코드 및 결과화면

```
문제 9번.py - C:\Users\Wbooti\Desktop\문제 9번.py (3.10.1)
File Edit Format Run Options Window Help
import turtle as t # 터틀 모듈을 불러오고 객체를 t라고 설정
t.shape() # 터틀 객체는 없는 것으로 함
i = 1 # 반복 숫자 초기화 여기서는 1로 초기화를 함
for i in range(11): # 10까지 반복
    t.down();t.circle(10*i);t.up() # 반지름을 10배수로 증가시키면서 원을 그림
```



9. 알고리즘

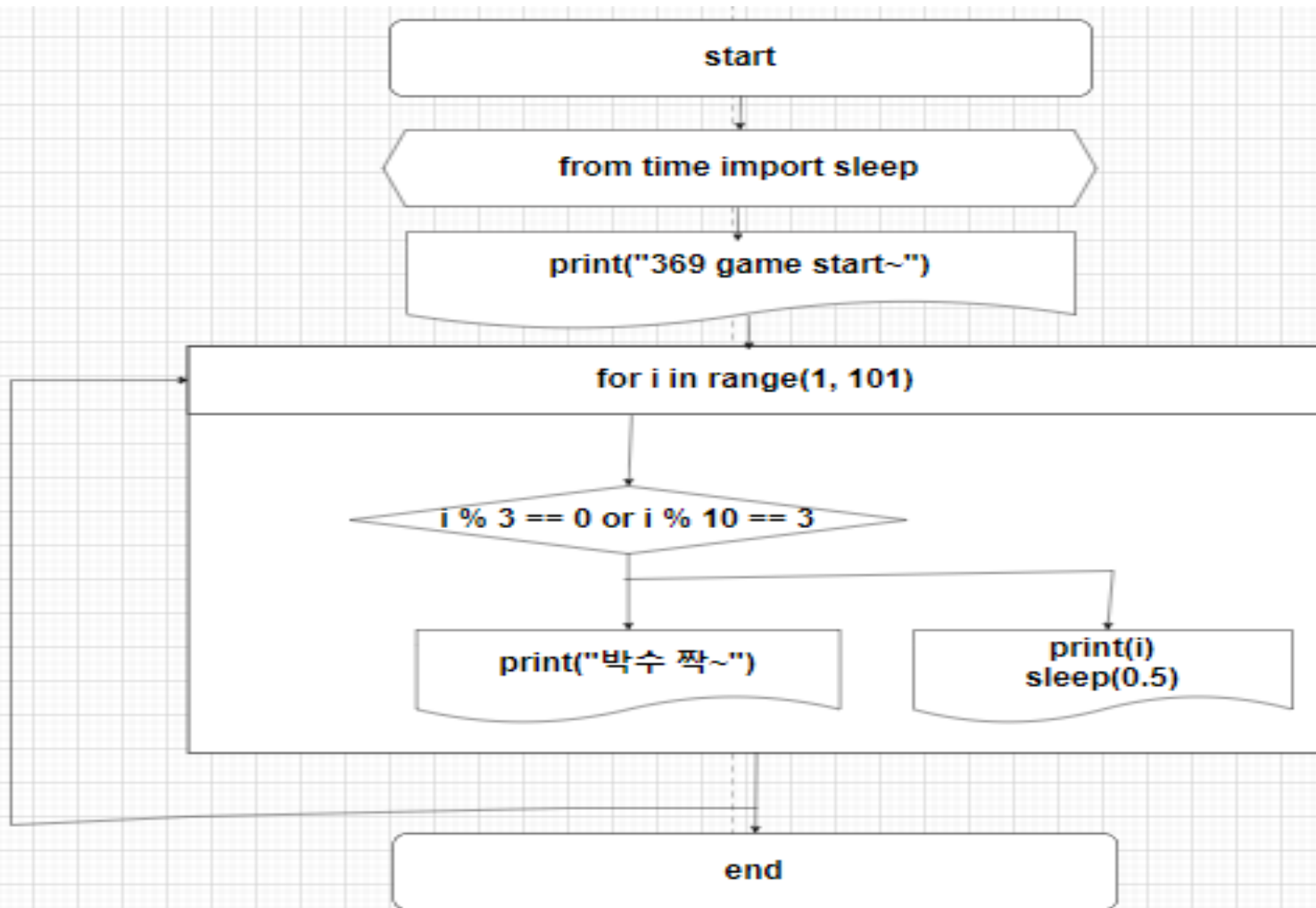


10. 소스 코드 및 결과화면

```
from time import sleep
print("369 game start~")
for i in range(1, 101): # 1 ~ 100까지 숫자가 반복되도록 함
    if(i % 3 == 0 or i % 10 == 3): # i가 3의 배수이거나 끝자리가 3으로 끝난다면
        print("박수 짹~")
        sleep(0.5) # 0.5초 동안 대기하다가 다음 숫자 출력
    else:
        print(i) # 박수 짹 ~ 을 제외한 나머지 숫자 출력
        sleep(0.5)
```

```
Python 3.8.0 (tags/v3.8.0:fa919fd, Oct 14 2019, 19:37:50) [MSC
v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more informati
on.
>>>
===== RESTART: C:\Users\Wbooti\Desktop
W문제 10번.py =====
369 game start~
1
2
박수 짹~
4
5
박수 짹~
7
8
박수 짹~
10
11
박수 짹~
```

10-1. 알고리즘



11. 소스코드 및 실행 화면

```
import turtle as t
from random import randint

t.shape("turtle") # 그래픽 객체는 turtle

s = t.textinput("", "이름을 입력하시오:") # 본인의 이름을 입력할 수 있는 창 생성

t.write("안녕하세요?" + s + "씨, 터틀 인사 드립니다.") # 본인의 이름 및 레이블이 생성

t.color("orange");t.speed(8);t.up();t.goto(-200, 200) # 오렌지 객체 이동
for step in range(20): # 터틀 경주를 할 경주선 그리기
    t.rt(90);t.down();t.fd(130);t.up();t.bk(130);t.lt(90);t.fd(25)

t.goto(0, -55) # 오렌지 객체 이동
t.down()
t.fd(50);t.lt(90);t.fd(20);t.lt(90);t.fd(100);t.lt(90);t.fd(20);t.lt(90);t.fd(50)
# 주황색 거북이가 사각형을 그리고 가운데 위치에 올 수 있도록 설정

t1 = t.Turtle() # 레드 객체 생성 및 위치 이동
t1.color('red');t1.shape('turtle');t1.penup();t1.goto(-220,170);t1.pendown()

t2 = t.Turtle() # 노랑 객체 생성 및 위치 이동
t2.color('yellow');t2.shape('turtle');t2.penup();t2.goto(-220, 140);t2.pendown()

t3 = t.Turtle() # 블루 객체 생성 및 위치 이동
t3.color('blue');t3.shape('turtle');t3.penup();t3.goto(-220,110);t3.pendown()

t4 = t.Turtle() # 그린 객체 생성 및 위치 이동
t4.color('green');t4.shape('turtle');t4.penup();t4.goto(-220, 80);t4.pendown()
```

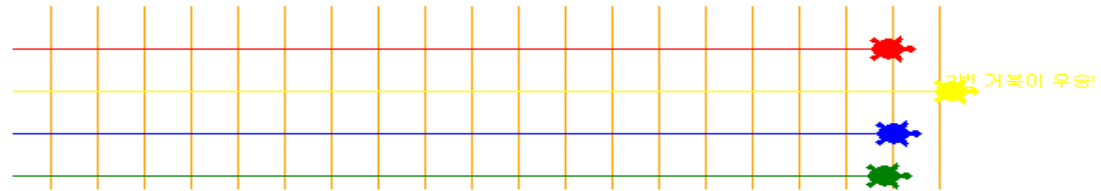
11-1. 소스코드 및 실행화면

```
ad1 = 0 # 1번 거북이가 이동한 거리
ad2 = 0 # 2번 거북이가 이동한 거리
ad3 = 0 # 3번 거북이가 이동한 거리
ad4 = 0 # 4번 거북이가 이동한 거리

for turn in range(180): # 179까지 1~5정도의 이동 수를 바탕으로 거북이들이 이동
    r1 = randint(1, 5) # 랜덤 모듈의 randint 메서드 사용 1 ~ 5까지의 난수 범위 설정
    r2 = randint(1, 5)
    r3 = randint(1, 5)
    r4 = randint(1, 5)
    t1.fd(r1)
    t2.fd(r2)
    t3.fd(r3)
    t4.fd(r4)
    ad1 += r1
    ad2 += r2
    ad3 += r3
    ad4 += r4

    if ad1 >= 500: # 1번 거북이 500 이상 이동
        t1.write("1번 거북이 우승!")
        break
    elif ad2 >= 500: # 2번 거북이 500 이상 이동
        t2.write("2번 거북이 우승!")
        break
    elif ad3 >= 500: # 3번 거북이 500 이상 이동
        t3.write("3번 거북이 우승!")
        break
    elif ad4 >= 500: # 4번 거북이 500 이상 이동
        t4.write("4번 거북이 우승!")
        break
```

11-2. 실행화면



안녕하세요?문성원씨, 터틀 인사 드립니다.



11.2 알고리즘

