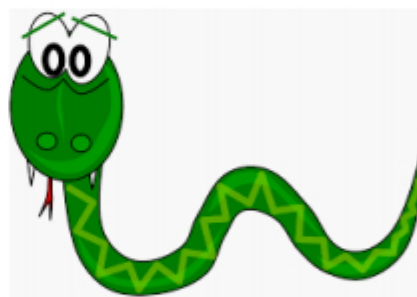
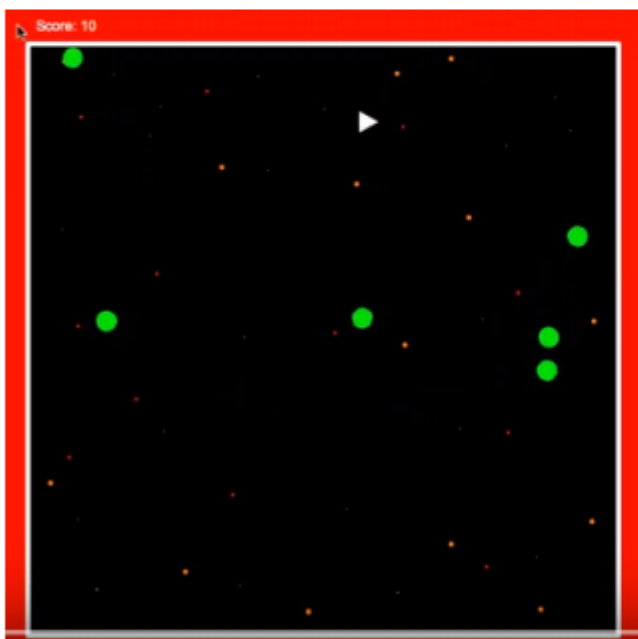
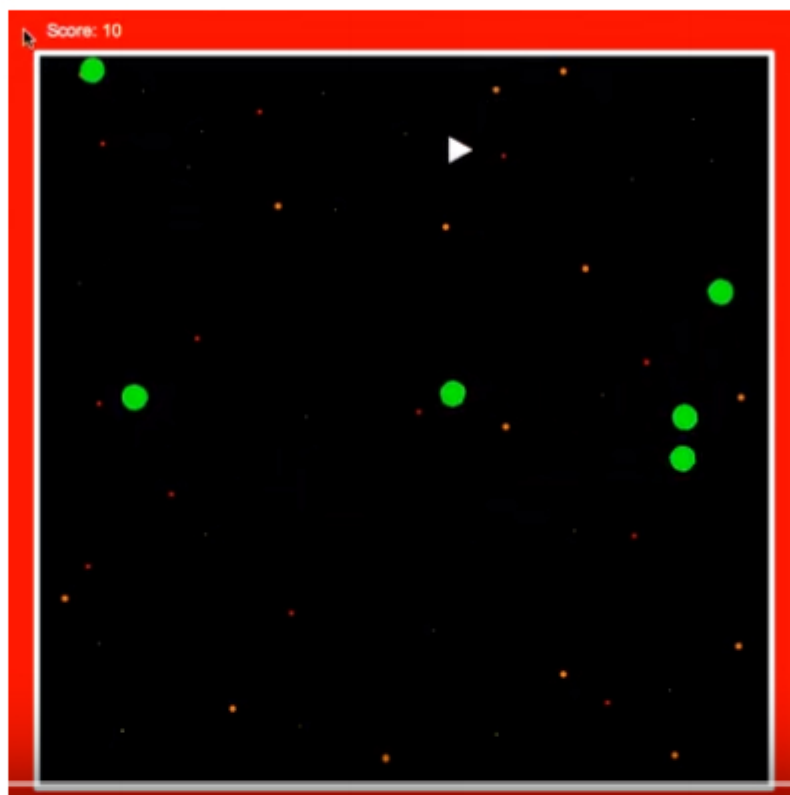


BÀI SỐ 8. LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG TRONG PYTHON (TIẾP)

- **Kiến thức:**
 - Ôn tập lại kiến thức lập trình hướng đối tượng: Class và Object
 - Ứng dụng lập trình hướng đối tượng vào lập trình ứng dụng
- **Kỹ năng:**
 - Làm việc với ngôn ngữ python, kỹ năng làm việc nhóm.
 - Luyện tập kỹ năng thuyết trình qua việc ôn tập cuối buổi học.
- **Sản phẩm:** Ứng dụng phương pháp lập trình hướng đối tượng làm Games “chiến binh không gian ver 3 với OOP”



1. Giới thiệu Game: Chiến binh không gian Ver 3







Kịch bản Game:

- Nhân vật: có 2 nhân vật đó là: quái vật (hình tròn) và siêu anh hùng (hình tam giác)
- Quái vật sẽ di chuyển ngẫu nhiên trong màn hình, nếu va chạm vào biên sẽ xuất hiện ở vị trí khác. Siêu anh hùng sẽ di chuyển thông qua sự điều khiển của người chơi.
- Siêu anh hùng sẽ di chuyển lao vào quái vật, mỗi lần lao trúng quái vật nào thì điểm số sẽ tăng lên và quái vật sẽ được thiết lập ở 1 vị trí khác.

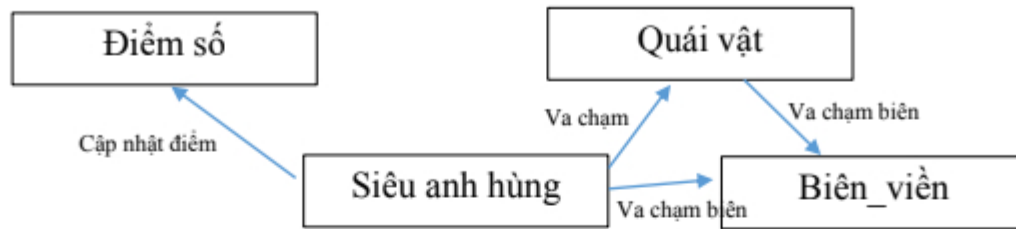
2. Ứng dụng lập trình hướng đối tượng vào Game

2.1. Phân tích ứng dụng Game theo hướng lập trình hướng đối tượng

Ứng dụng game trên có 4 đối tượng sẽ tương tác với nhau:

<p>+ Siêu anh hùng:</p> 	<p><u>Siêu anh hùng:</u></p> <p>- Thuộc tính:</p> <ul style="list-style-type: none"> + Hình dạng, màu sắc + Tốc độ <p>- Hành vi:</p> <ul style="list-style-type: none"> + Tiến , lùi + Quai trái, phải
<p>+ Quái vật:</p> 	<p><u>Quái vật:</u></p> <p>- Thuộc tính:</p> <ul style="list-style-type: none"> + Hình dạng, màu sắc + Tốc độ <p>- Hành vi: di chuyển</p>
<p>+Điểm số:</p> 	<p><u>Điểm:</u></p> <p>- Thuộc tính:</p> <ul style="list-style-type: none"> + Hình dạng, màu sắc <p>- Hành vi: cập nhật điểm</p>
<p>+ Biên viên:</p> 	<p><u>Biên viên:</u></p> <p>- Thuộc tính:</p> <ul style="list-style-type: none"> + Hình dạng, màu sắc <p>- Hành vi: Xác định va chạm biên</p>

Tương tác giữa các đối tượng:



2.2. Giới thiệu mã nguồn:

Lớp đầu tiên mà chúng ta sẽ làm đó là tạo ra lớp **siêu anh hùng** với:

Các thuộc tính:

- + Hình dạng
- + Màu sắc
- + Tốc độ di chuyển

Hành vi là:

- + Di chuyển
- + Di chuyển theo các phím



Tạo lớp Class sieu_anh_hung

```

# Chèn thư viện đồ họa turtle
import turtle
import math
import random

# Khởi tạo màn hình đồ họa
screen = turtle.Screen()

# thiết lập độ rộng của màn hình đồ họa
screen.setup(400, 400)

#thiết lập hình ảnh nền cho màn hình đồ họa
screen.bgcolor("red")

#Tựa đề của số
  
```

```
screen.title("Giải cứu không gian 3")

#--- Class sieu_anh_hung---
class sieu_anh_hung(turtle.Turtle):
    # hàm khởi tạo các thuộc tính mỗi khi class được gọi
    def __init__(self): # chú ý: __init__ không phải là: _init_ nhé
        turtle.Turtle.__init__(self)
        self.penup()
        self.speed(0)
        self.shape("triangle") #thuộc tính hình dạng
        self.color("white") # thuộc tính màu sắc
        self.speed = 1 # thuộc tính tốc độ

    def move(self):
        self.forward(self.speed)

    def turnleft(self):
        self.left(30)

    def turnright(self):
        self.right(30)

    def accelerate(self):
        self.speed += 1

    def decelerate(self):
        self.speed -= 1

# khởi tạo đối tượng của lớp siêu anh hùng
player = sieu_anh_hung()
```

```
# thiết lập điều khiển nhân vật bằng bàn phím
```

```
turtle.listen()
```

```
turtle.onkey(player.turnleft,"Left")
```

```
turtle.onkey(player.turnright,"Right")
```

```
turtle.onkey(player.accelerate,"Up")
```

```
turtle.onkey(player.decelerate,"Down")
```

```
#-----Chương trình chính-----
```

```
while True:
```

```
    player.move()
```

Để giới hạn phạm vi di chuyển, chúng ta bổ sung lớp **Class Bien_vien**:

Tạo lớp: Class Bien_vien

```
#-----CLASS BORDER-----
```

```
class Bien_vien(turtle.Turtle):
```

```
    def __init__(self): # chú ý: __init__ không phải là: _init_ nhé
```

```
        turtle.Turtle.__init__(self)
```

```
        self.penup()
```

```
        self.speed(0)
```

```
        self.color("white") # thuộc tính màu sắc
```

```
        self.pensize(5) #màu biên
```

```
    def draw_border(self):
```

```
        self.penup()
```

```
        self.goto(-300, -300)
```

```
        self.pendown()
```

```
        self.goto(-300,300)
```

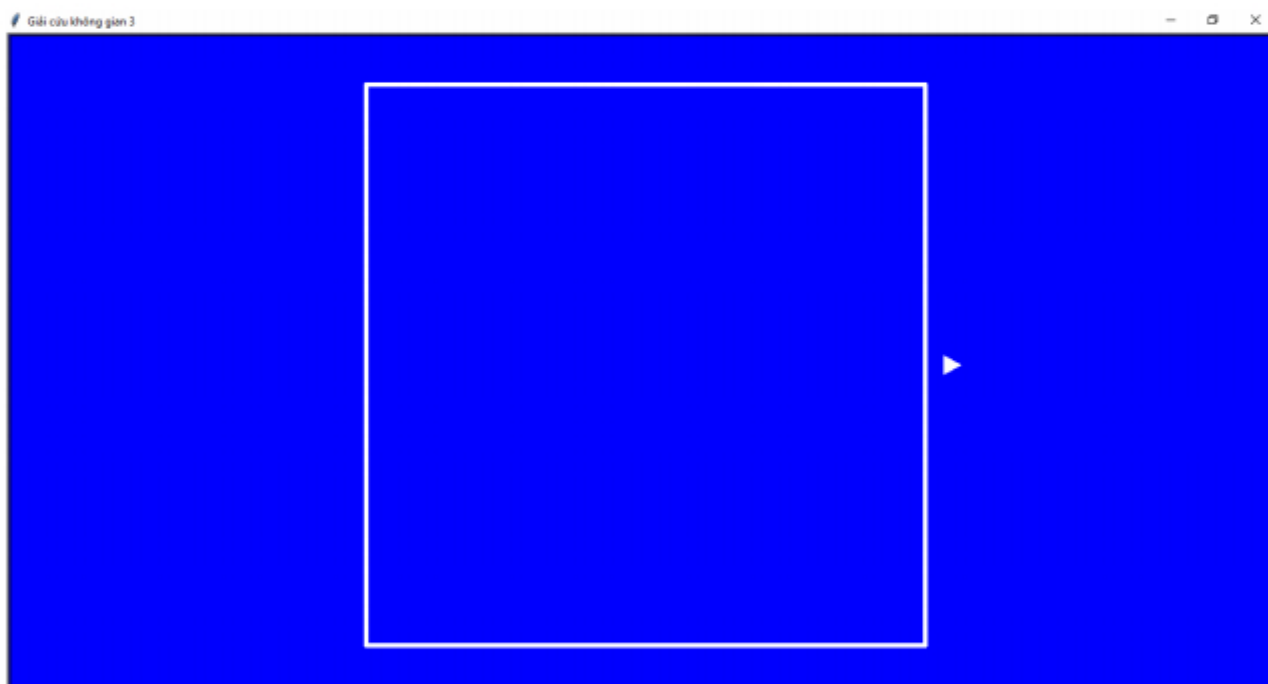
```
self.goto(300,300)  
self.goto(300,-300)  
self.goto(-300,-300)  
self.hideturtle()
```

Khởi tạo đối tượng biên viên

```
border = Bien_vien()
```

```
border.draw_border()
```

Kết quả:



Khung viền đã xuất hiện, tuy nhiên đối tượng Siêu anh hùng khi di chuyển đã vượt biên ra bên ngoài (như hình trên). Vậy ta sẽ phải bổ sung code cho phần Class của siêu anh hùng chỉ được phép di chuyển trong giới hạn Biên thôi. Nếu gặp biên sẽ phải quay lại. Các em chỉnh sửa lại hàm **move()** của lớp **siêu_anh_hung()** nhé:

Sửa lại hàm move() trong lớp siêu_anh_hung()

def move(self):

self.forward(self.speed)

Kiểm tra khi siêu anh hùng va chạm vào biên viên, khi va chạm vào biên sẽ quay phải 60 độ.

if self.xcor() > 290:

self.setx(290)

self.right(60)

if self.xcor() < -290:

self.setx(-290)

self.right(60)

if self.ycor() > 290:

self.sety(290)

self.right(60)

if self.ycor() < -290:

self.sety(-290)

self.right(60)

Như vậy, nhân vật siêu anh hùng với các tính năng cơ bản đã hoàn thành, bây giờ chúng ta sẽ chuyển sang nhân vật Quái vật với:

Quái vật:**- Thuộc tính:**

+ Hình dạng, màu sắc

+ Tốc độ

- Hành vi: Di chuyển

Class Quái vật

class Quai_vat(turtle.Turtle):**def __init__(self):** *# chú ý: __init__ không phải là: _init_ nhé*

turtle.Turtle.__init__(self)

self.penup()

self.speed(0)

 self.shape("circle") *#thuộc tính hình dạng* self.color("green") *# thuộc tính màu sắc* self.speed = 3 *# thuộc tính tốc độ*

self.goto(random.randint(-250, 250),random.randint(-250, 250))

self.setheading(random.randint(0,360))

*#Hàm thiết lập khi va chạm với siêu anh hùng thì quái vật sẽ đặt ở 1 vị trí ngẫu nhiên***Def jump(self):**

self.goto(random.randint(-250, 250),random.randint(-250, 250))

self.setheading(random.randint(0,360))

*# Hàm di chuyển, khi quái vật chạm viên sẽ quay lại***def move(self):**

self.forward(self.speed)

if self.xcor() > 290:

self.setx(290)

self.right(60)

if self.xcor() < -290:

```
self.setx(-290)
self.right(60)
if self.ycor() > 290:
    self.sety(290)
    self.right(60)
if self.ycor() < -290:
    self.sety(-290)
    self.right(60)
```

Khai báo đối tượng quái vật

quaivat= Quai_vat()

Các em bổ sung vào phần while True:

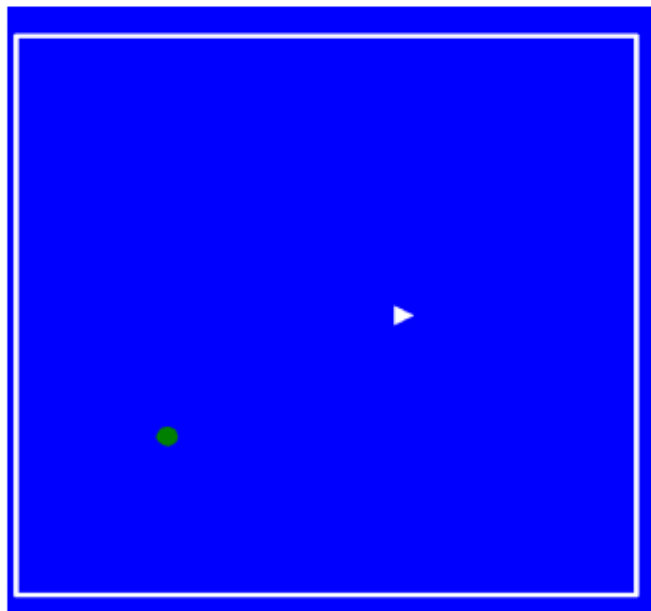
#-----Chương trình chính-----

while True:

player.move()

quaivat.move() #bổ sung hàm này để cho quái vật chuyển động

Kết quả:



Tiếp theo, chúng ta viết hàm kiểm tra sự va chạm của quái vật và siêu anh hùng. Hàm này, thầy đã giới thiệu trong bài 6 rồi, các em xem lại nhé.

Hàm kiểm tra sự va chạm của quái vật và siêu anh hùng

```
def isCollision(t1, t2):
```

```
    # khoảng cách giữa siêu anh hùng và quái vật
```

```
    d = math.sqrt(math.pow(t1.xcor() - t2.xcor(),2) + math.pow(t1.ycor() - t2.ycor(),2))
```

```
    if d<20:
```

```
        return True
```

```
    else:
```

```
        return False
```

Để làm tăng tính hấp dẫn của game, chúng ta hãy tạo ra nhiều quái vật:



Tại code khai báo đối tượng quái vật, các em chỉ cần sửa lại như sau:

```
quaivat= Quai_vat()
```

⇒ **Sửa lại thành:**

```
quaivat=[]
```

```
for i in range(6):
```

```
    quaivat.append(Quai_vat())
```



⇒ Sửa lại vòng lặp While True thành:

Chương trình chính:**while True:**

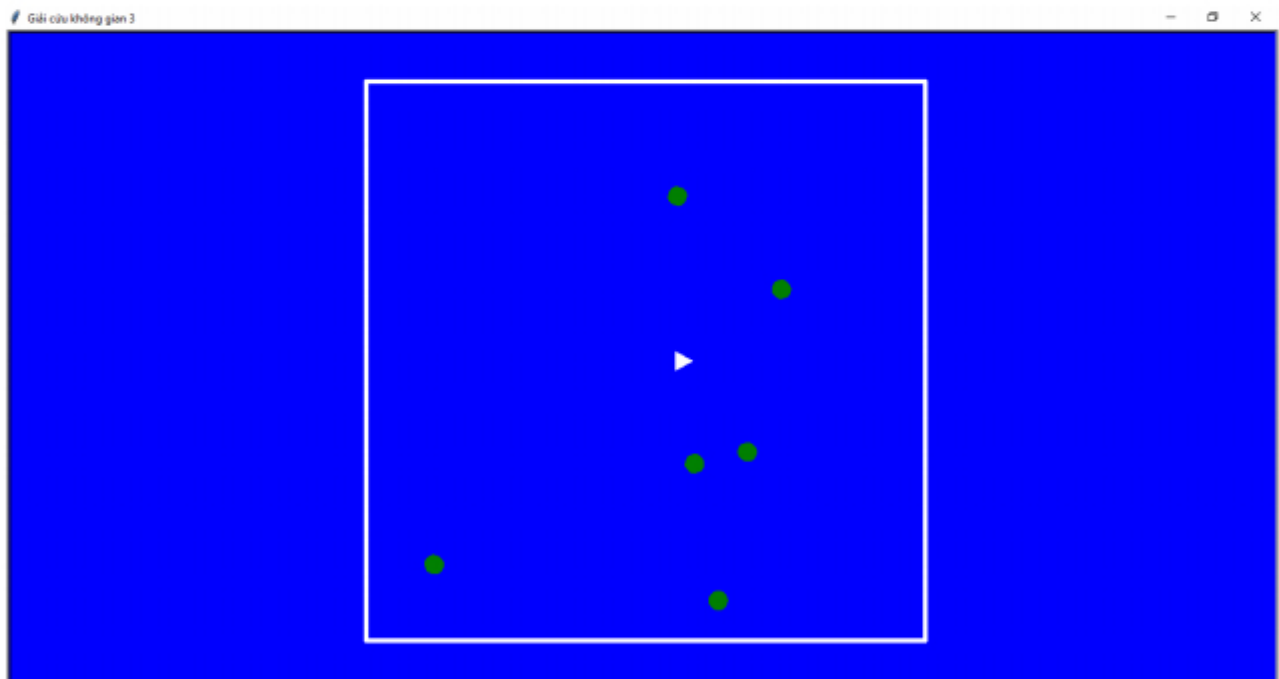
player.move()

*# Thiết lập 6 con quái vật di chuyển***for qv in quaivat:**

qv.move()

*# kiểm tra sự va chạm quái vật và siêu anh hùng***if isCollision(player,qv):**

qv.jump()

Kết quả:

Với kết quả trên, khi siêu anh hùng va chạm vào quái vật, thì quái vật đã biến mất và chuyển sang vị trí khác. Trong phần tiếp theo, chúng ta sẽ bổ sung tính năng cập nhật điểm mỗi khi siêu anh hùng va chạm vào quái vật.

Chúng ta sẽ tạo ra lớp điểm với:

- Thuộc tính:
 - + Màu chữ
 - + Kiểu chữ
 - + Loại chữ
- Hành vi: Cập nhập điểm



Classe Điểm

class Score(turtle.Turtle):

```
def __init__(self): # chú ý: __init__ không phải là: _init_ nhé
```

```
    turtle.Turtle.__init__(self)
```

```
    self.penup()
```

```
    self.hideturtle()
```

```
    self.speed(0)
```

```
    self.color("white")
```

```
    self.goto(-290,310)
```

```
    self.score =0
```

def update_score(self):

```
    self.clear()
```

```
    self.write("Score: {}".format(self.score), False, align="left", font=("Arial",14))
```

def change_score(self, points):

```
    self.score += points
```

```
    self.update_score()
```

khởi tạo đối tượng của lớp Score()

score=Score()



Để cập nhật được điểm, chúng ta sửa lại phần while true:

Chương trình chính:**while True:**

player.move()

thiết lập 6 con quái vật di chuyển

for qv in quaivat:

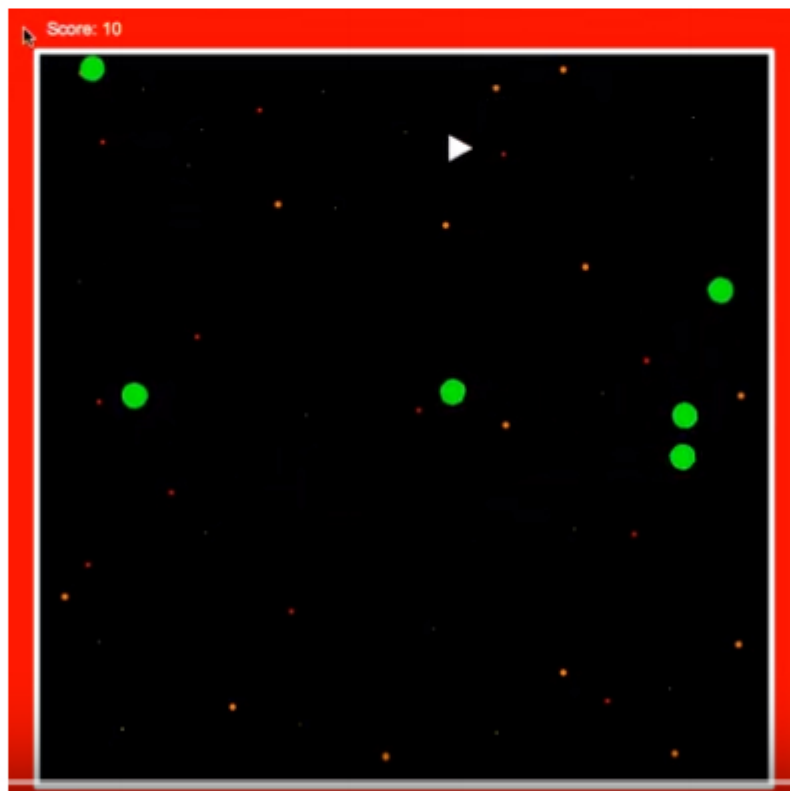
qv.move()

kiểm tra sự va chạm quái vật và siêu anh hùng

if isCollision(player,qv):

qv.jump()

score.change_score(10) # cập nhật điểm

Kết quả:



TỔNG KẾT BÀI HỌC

Xin chúc mừng các em đã hoàn thành bài học số 8, trong bài này, các em cần chú ý những nội dung sau:

1. Viết kịch bản game trước khi lập trình
2. Phân tích theo hướng lập trình hướng đối tượng
3. Sử dụng các lớp và đối tượng để lập trình

The Next Step



Nội dung bài học tiếp theo: Chúng ta sẽ ôn tập toàn bộ nội dung đã học từ bài 1 đến bài 9. Và chuẩn bị ý tưởng cuối khóa.

Các em hãy tìm hiểu trước nhé!



BÀI TẬP VỀ NHÀ

Các em hãy trình bày ý tưởng của mình về 1 chương trình định làm:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....



