**Computational Science Project (SC313101) : การวิเคราะห์ AI เกมงูกินหางว่าจะทำคะแนนในรูปแบบที่อาหารสุ่มเกิดแบบ random uniform distribution กับแบบ Clumsy รูปแบบใดจะทำให้ AI ที่สร้างทำคะแนนเยอะที่สุด**

**สมาชิกในกลุ่ม (Team members):**

**1) ชื่อ นายศุภพงศ์ สกุลคู รหัสนักศึกษา 5930215910-7**

**2) ชื่อ นายณัฐวุฒิ จันทกุล รหัสนักศึกษา 613021054-7**

**3) ชื่อ นายจักรพงศ์ ฤาชา รหัสนักศึกษา**

# Project Proposal:

## เนื้อหารายงานเสนอโครงงาน มีรายละเอียดดังนี้

## 1. ความเป็นมาของโครงงาน

เนื่องจากเกมงูกินหางเป็นเกมที่นิยมมาอย่างยาวนานในหมุ่เกม 2d เพื่อเพิ่มscore ในการเล่นให้กับผู้เล่น ทางผู้จัดทำจึงอยากทดสอบว่าการกระจายตัวของอาหารแบบกลุ่ม cluster กับแบบ random uniform distributiom อย่างไหนงูจะกินอาหารได้มากกว่ากัน (ได้ score มากกว่ากัน) เพื่อให้ผู้เล่นเลือกได้ว่าถ้าได้ไปเล่นเกมงูกินหางจริงๆจะเล่นเกมในโหมดที่อาหารกระจายตัวแบบ cluster หรืออาหารกระจายตัวแบบ random uniform distribution เพื่อให้ได้ score สูงสุด

## 2.ขั้นตอนการสร้าง AI เกมงูกินหาง

## 2.1 ติดตั้ง module pygame โดย run คำสั่ง pip install pygame ใน cmd (command prompt)

## 

## ถ้าติดตั้งไว้อยู่แล้วจะได้ผล ดังภาพ

## 2.2 ให้ import module ที่ต้องใช้ ตาม Code ด้านล่าง

## 

## อธิบายเพิ่ม

## Module ที่ใช้ในนี้สำคัญๆ มี 4 ตัว ได้แก่

## 1. pygame สำหรับ สร้างเกม

## 2. tkinter สำหรับใช้สร้าง gui ต่างๆในเกม เช่น หน้าต่างเกม

## 3. random เพื่อสุ่มค่า int ในขั้นตอนการ generate food

## 4. time เอาไว้จับเวลา

## 2.3 สร้าง Array เก็บสีต่างๆที่ใช้ในเกม และกำหนด Board ให้กับเกม โดยตัวเลขต่างๆใน Board มีความหมาย ดังนี้

## 2.3.1 เลข 0 คือ ช่องว่าง

## 2.3.2 เลข 1 คือ Body หรือตัวงู

## 2.3.3 เลข 2 คือ อาหาร

## 2.3.4 เลข 3 คือ หัวงู

## 2.3.5 เลข 4 คือ Block ชนแล้วงูตาย

## ซึ่งก็จะเขียน code ได้ตามภาพ

## 

## HEIGHT คือ ความสูง มีค่า 14

## WIDTH คือ ความกว้าง มีค่า 18

## 2.4 กำหนดตัวงูเริ่มต้นโดยสร้าง array ชื่อ default\_location เป็น array ของ tuple โดยให้สมาชิกตัวแรกของ array คือ หัวงู (Head) จะเขียน Code ได้ตามภาพ

## 

## 2.5 เซตจุดที่อาหารโผล่ขึ้นตอนแรก เป็น จุด (5,1)

## 

## 

## 2.6 ทำการเซต parameter ให้ GUI และตัวควบคุมซ้าย ขวา บน ล่าง (Controller) ดังนี้

## 

## อธิบาย x[1]= ตำแหน่งถัดไป , x[0]= ตำแหน่งปัจจุบัน

## 2.7 สร้าง Class Snake โดยจะอธิบายเป็นลำดับในแต่ละ method ดังนี้

## 2.7.1 กำหนดค่าเริ่มต้นให้กับ Class Snake ด้วย def \_\_init\_\_ ดังนี้

## 

## อธิบาย

## 1. self.locations จะรับต่า default\_location ที่เราเซตไว้แต่แรกนอก Class Snake เข้ามา ซึ่งก็คือ ตำแหน่งของตัวงูตอนเริ่มเกมตอนแรก

## 2. self.speed =0 คือ speed ตอนเริ่มเกมของงูเป็น 0 ยังไม่ได้ทำอะไร

## 3.self.board = board ก็คือ ค่า board ที่เราเซตไว้นอก Class Snake pass เข้ามาในตัวแปร self.board ถ้าอธิบายให้เห็นภาพเหมือนกับการเซตด่านให้กับ ai งู คือ เซตบล็อกสิ่งกีดขวางต่างๆให้กับตัวงู

## 4. self.score =0 คือ score เริ่มแรกเป็น 0

## 5. self.virtual = False

## 2.7 ทำการสร้าง method head , tail , move เพื่อควบคุมตัว ai

## 

## อธิบาย

## 1. def head(self): จะ return ตำแหน่งหัวงูออกมาก็คือ บล็อกแรก (self.locations[0])

## 2. def tail(self): return ตำแหน่งหางงูออกมา ก็คือ ตำแหน่งบล็อคสุดท้าย(self.locations[-1])

## 3. def move ควบคุมการเดินของงู โดยมีเงื่อนไขว่า

## If ถ้า self.head() == food : ก็คือ ถ้าตำแหน่งหัวงูไปตรงกับตำแหน่ง food อาหาร ให้งูเพิ่มบล็อค 1 บล็อค ก็คือ หางงอก 1 บล็อก ( จาก Code : self.locations=self.locations+[temp] ) แล้วให้ generate food ใหม่ และให้ score +1

## If not self.virtual -> if not False ให้โปรแกรมหยุดเป็นเวลาเท่ากับ self.speed

## จากนั้นเมื่อเช็ค if เสร็จแล้วให้ทำการอัปเดต Board ด้วยคำสั่ง self.update\_board() และสิ้นสุด method move

## 2.8 สร้าง method update\_board กับ method generat\_food เพื่อสร้างอาหารขึ้นใหม่เมื่องูได้กินอาหารเข้าไป

## 

## 2.9 ให้วิธีเดินแก่งู

## 

## 2.10 สร้าง method draw เป็น method ที่วาดเกมงูกินหาง สี บล็อก งู ทั้งหมดออกมาเป็นรุปร่างเกม

## 

## Canvas.update() อัปเดตหน้าต่างเกม

## 2.11 สร้าง method อื่นๆดังนี้

## 

## 

## 2.12 Main ไว้ Run Program

## 

## อธิบายคำสั่ง

## if \_\_name\_\_ ==’\_\_main\_\_’ หมายถึง Main root = Tk() สร้าง จอ (Screen) มา

## Root.title(‘Snake AI’) คือ label จอเกมว่า Snake AI

## Canvas = Canvas(root,bg=”white”) ให้กรอบจอเกมสีขาว

## S= Snake(Board) สร้าง object S ขึ้นจาก Class Snake และ pass BOARD เข้าไป แล้วเรียนใช้ method run ซึ่งเป็นคุณสมบัติของ object s เพื่อ run เกมงู

## ตัวอย่างภาพการ Run AI Snake บน Python Script

## 

## ตายแล้วมี Score แสดงผลบน Python Script ดังภาพ

## 

## ภาพตอนงูตายและเกมจบ

**3. หลักการที่ใช้**

1. **กฎ 95 เปอร์เซ็นต์ อธิบายว่า ข้อมูล 95% ของข้อมูลทั้งหมด จะอยู่ในช่วง (mean - 2*SD , mean+2*SD)**

## 

## 2. จะใช้กฎ 95% เพื่อทำการ Simulation เกม 100 ครั้ง จากค่าเฉลี่ยน และ SD ของการรันเกม 10 ครั้ง ในแต่ละรูปแบบที่ Ai งู ที่สร้างเดิน ได้แก่ แบบที่อาหารกระจายแบบ random uniform และแบบที่อาหารกรจายตัวแบบเป็นกลุ่ม

## 4. วิธีการดำเนินงานในรายละเอียดของการวิเคราะห์ AI ในรูปแบบการเดินที่อาหารกระจายตัวแบบ random uniform กับแบบที่อาหารรวมกลุ่ม

## 1. สร้าง Python Script 2 ตัว โดยตัวแรก ให้ออกแบบโดยให้ อาหารกระจายตัวแบบ random uniform distribution แล้วrun AI บนนั้น กับอีกแบบคือ อาหารรวมกลุ่มกัน Clumsy แล้วให้ AI run บนนั้น อย่างละ 10 รอบ

## 2. ได้ข้อมูล score และเวลาในการ run ของ AI ในทั้ง 2 รูปแบบ อย่างละ 10 รอบ

## 3. หาค่าเฉลี่ยและ SD ของ score ทั้ง 10 รอบ ที่ AI งูได้ ใน ทั้ง 2 รูปแบบ

## 4. ทำการ Simulation การ run ของ AI งู 100 ครั้ง โดยการสุ่ม random ค่าจากข้อมูลทั้งหมด โดยใช้กฎ 95% คือ สุ่มค่าในช่วง (mean-2SD,mean+2SD) มา 100 รอบ เก็บไว้ใน array แล้วนำมา plot histogram โดยแกน x เป็น score ที่ได้ และแกน y เป็นความถี่ที่ score นั้นออก ทำกับทั้งรูปแบบ Random uniform distribution food กับ Clumsy Food

## 5. เปรียบเทียบว่ารุปแบบไหนให้ score ที่สูงกว่ากัน (ไม่มีเวลามาเกี่ยวข้อง)

## 

## Distribution

## 

## Clumsy

## สรุป จากกราฟ Histogram พบว่า ในรูปแบบที่อาหารกระจายตัวแบบ Random Uniform Distribution มีช่วง score ที่ AI ทำได้ตั้งแต่ 173 จนถึง 188 และมีความถี่มาก 165-172 และ 174-180 และ 182-184 ซึ่งมี score มากกว่า รูปแบบการเดินที่อาหารเป็นกลุ่ม Clumsy ที่ score อยู่ในช่วง 25-65 และมี score มีความถี่สูงช่วง score 30-65

## แต่ถ้าในเวลาจำกัด Clumsy จะได้ score มากกว่า Random Uniform Distribution