Week 3 Breakout Group 3

Max Fleming, Brendan Lienau, Michael Nadasky

# AI Navigation

SetWalkDirection(Direction dir)

walkingDirection = dir

End SetWalkingDirection

MoveInDirection()

Moves forward in walkingDirection

End MoveInDirection

Navigate()

SetWalkDirection(east)

MoveInDirection() until reaching door

open door

SetWalkDirection(north)

MoveInDirection() until reaching Harrison

SetWalkDirection(east)

MoveInDirection() until reaching 600

SetWalkDirection(south)

MoveInDirection() until reaching door

open door

End Navigate

# Snowman Drawing

* Three circles DONE
* Top hat DONE
* Three buttons on the middle circle DONE
* Carrot nose DONE
* Two Eyes DONE
* Two sticks for arms DONE
* Mouth Done

DrawALine(Float: StartX, Float: StartY, Float: EndX, Float: EndY, Color: color)

Draws a line from point (StartX, StartY) to point (EndX, EndY) that’s colored color

End DrawALine

DrawACircle(Float: Radius, Float: CenterX, Float: CenterY, Color: color)

Draws a circle with radius Radius centered at the point (CenterX, CenterY) that’s colored color

End DrawACircle

MakeASnowman(Float: X, Float: Y, Integer: Scale, Integer: DotSize)

//draws the three circles of the snowman

Float: Height = Y+ (Scale \* 13)

Float: MidCircleCenter = Y + (Scale \* 8)

Float: FirstCircleTop = Y + (Scale \* 6)

DrawACircle(Scale \* 3, X, Y + (Scale \* 3), white)

DrawACircle(Scale \* 2, X, MidCircleCenter, white)

DrawACircle(Scale, X, Y + (Height - Scale), white)

//draws the three buttons on the center circle

DrawACircle(DotSize, X, Y + (Scale \* 9), black)

DrawACircle(DotSize, X, Y + (Scale \* 10), black)

DrawACircle(DotSize, X, Y + (Scale \* 11), black)

//draws the top hat

DrawALine(X-(Scale \* 2), Height, X+(Scale \* 2), Height, black)

DrawALine(X-Scale, Height, X-Scale, Height+Scale, black)

DrawALine(X+Scale, Height, X+Scale, Height+Scale, black)

DrawALine(X-Scale, Height+Scale, X+1, Height+Scale, black)

//draws the nose

DrawALine(X, Height-Scale, X+(Scale/2), Height-Scale, orange)

//draws the eyes

DrawACircle(DotSize, X-(Scale/2), Height-(Scale/2), black)

DrawACircle(DotSize, X+(Scale/2), Height-(Scale/2), black)

//draws the arms

DrawALine(X-(Scale\*2), MidCircleCenter, X-(Scale\*3), FirstCircleTop, black)

DrawALine(X+(Scale\*2), MidCircleCenter, X+(Scale\*3), FirstCircleTop, black)

//draws the mouth

DrawALine(X-(Scale/2), Y+(Scale \* 12.25), X+(Scale/2), Y+(Scale \* 12.25), black)

End MakeASnowman

# Algorithm Analysis

Both algoritms take the same amount of time regardless of the preconditions; O(1). Navigate() always follows the same route (916 and 600 aren’t getting any further away from each other) and MakeASnowman() always makes the same snowman but bigger or smaller and in different places.

# Advanced Challenge

## 1.

Integer: CountDolls(Integer: Dolls[])

Integer: Count = 0

For i = 1 to length of Dolls

Count += Dolls[i]

End For

Return Count

End CountDolls

Best case: O(n)

Worst case: O(n)

## 2.

Integer[]: SortDolls(Integer: Dolls[])

Integer: SortedDolls[length of Dolls] = Dolls[]

For i = 1 to length of SortedDolls

For j = 1 to length of SortedDolls

If SortedDolls[j] > SortedDolls[j+1]

Integer: Temp = SortedDolls[j+1]

SortedDolls[j+1] = SortedDolls[j]

SortedDolls[j] = Temp

End If

End For

End For

Return SortedDolls

End SortDolls

Best case: O(n^2)

Worst case: O(n^2)

## 3.

Struct Doll

Boolean: IsCracked

Doll: NestedDolls[]

End Doll

Doll: FindCrackedDoll(Doll: Dolls[])

For i = 1 to length of Dolls

Doll: Temp = Dolls[i]

If Temp isCracked

Return Temp

Else

For j = 1 to length of Temp

If Temp[j] isCracked

Return Temp[j]

End If

End For

End If

End For

Return Null

End FindCrackedDoll

Best runtime: O(n)

Worst runtime: O(n)