Hope IT - Python course - updated on June 2023:

All the code is in the below GitLab site. These code in these lesson plans are in the “examples” folder:

https://gitlab.com/hope_it/auto-bot-maze

Week 1:

Objectives:

1) VS Code:
    a) Get to know the environment.
    b) How to open a file.
    c) How to save a file.
2) Python code:
    a) Hello World
    b) Variables
    c) DEF method
    d) IF statements
    e) FOR loop.

Introductions: Name, school year (8th grade, Freshman, etc), and school

Opening Prayer

Open VS Code:

- Review Functions: add.py

```
def add (a, b):
    c = a + b
    return c

print(add(1,1))
print(add(2,2))
```

- Review Loops and Conditions: birthday.py

```
age = 17

test = 0
while test < age:
```

```
            test = test + 1
            if test == 16:
                print("Happy Birthday, you can drive now!")
            else:
                print("Happy Birthday!")

    print("All done.")
```

- Open manual.py
- Review the code. Don't need to understand it, just see there are instructions. We will run it soon.

Scriptural Reflection:

With solving mazes, it's like being lost. Let's review in Luke 15, what Jesus says about being lost, and then found. There are 3 parables here. Let's first review what Jesus says about what a parable is.

What is parable..in Jesus' explanation:

Luke 8:

8 When he said this, he called out, "Whoever has ears to hear, let them hear."

9 His disciples asked him what this parable meant. **10** He said, "The knowledge of the secrets of the kingdom of God has been given to you, but to others I speak in parables, so that,

"'though seeing, they may not see;

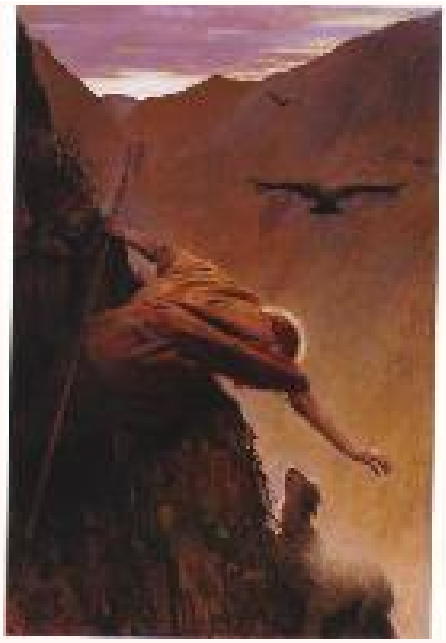
though hearing, they may not understand.'[a]

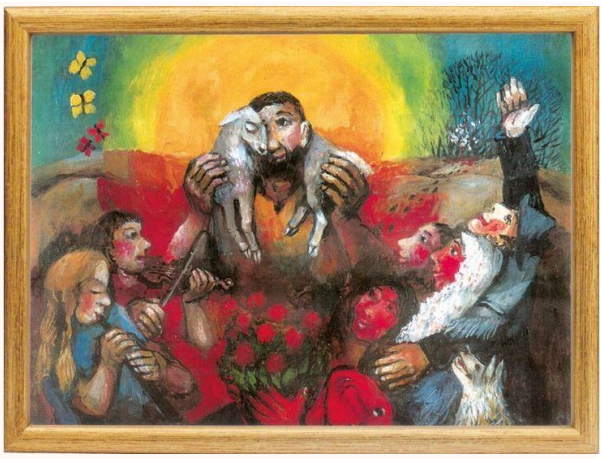
Luke 15

1 Now the tax collectors and sinners were all drawing near to hear him. **2** And the Pharisees and the scribes murmured, saying, "This man receives sinners and eats with them."

3 So he told them this parable: **4** “What man of you, having a hundred sheep, if he has lost one of them, does not leave the ninety-nine in the wilderness, and go after the one which is lost, until he finds it? **5** And when he has found it, he lays it on his shoulders, rejoicing. **6** And when he comes home, he calls together his friends and his neighbors, saying to them, ‘Rejoice with me, for I have found my sheep which was lost.’ **7** Just so, I tell you, there will be more joy in heaven over one sinner who repents than over ninety-nine righteous persons who need no repentance.

Show some art work:

	The Good Shepherd 1900 by Alfred Usher Soord (1868-1915) Oil on Canvas 275 x 182 cm

	Rietje Bakker

Week 2:

Objectives:

3) Use what we learned from Week 1, and use them to solve a maze.
4) Python code:
    a) Create individual Move instructions.
    b) FOR loop.
    c) Create a Function

Opening Prayer

Solve a maze manually:

- Have the kids use the arrow keys (up, down, left, right) to solve the maze.
- Have them do it again, but this time write down the steps (doing it again means invoking the script with --seed=NNNNNNN, where the value is what was output from the earlier run):
    - N (North) - N - W (West) - W - W - S (South) - S - S (for example)

Let's enter Python code to have the virtual robot follow instructions we used in the manual example:

- Open the scripted.py file in VS Code. We will build the instructions there.
- We will have the kids type in something like the following. The key is each line is one step.
- Have them save their work...don't want to lose work!
- Use Bash to run the code.
- We can run it after 2 steps, 4 steps, just to test. See if it's working and continue until the robot solves the maze. We won't focus on the fact that "`moves`" is a list just yet.

The following list of moves will work for maze generated by `--seed 2613097548 --size S`

```
moves = []
moves.append('E')
moves.append('E')
moves.append('E')
moves.append('S')
moves.append('S')
moves.append('S')
moves.append('S')
moves.append('S')
moves.append('S')
moves.append('S')
moves.append('S')
moves.append('W')
moves.append('W')
moves.append('W')
```

- Ignore the below code. We'll get to that later.

```
def callback(position):
    time.sleep(0.5)
    global moves
    if len(moves) == 0:
        time.sleep(5)
        move = 'Q'
    else:
        move = moves.pop(0)
    return move

if __name__ == '__main__':
    app = Runner(callback)
    app.run()
```

- *BREAK: Somewhere here, we are probably around an hour into the course. We should stop and have everyone get up and take a mental break. Get more orange juice, etc.*

- Now, why do we have to write each statement? If our robot has to move East 3 steps, why don't we just say do that all once? Let's learn the FOR loop. Do each one at time, save and test. We'll need to explain the FOR loop, and what each part is, see the one yellow highlighted part. You can do this on the white board. You will have to explain the `i` variable, and why we start with zero. Maybe write down each step as the loop works for each iteration...that's probably best. (And we'll discover it's doing the same thing as our long list of instructions, just saving the programmer time in coding it this way.)
- Note: We can either save our previous work, or comment it out. Maybe test to comment it out, so they can easily reference it, to see how many steps it took in the first round.

```
moves = []
for i in range(0, 3):
    moves.append('E')
for i in range(0, 8):
    moves.append('S')
for i in range(0, 3):
    moves.append('W')
```

- This is the final piece. The FOR loop was nice, but why do we have to write so many FOR loops? There really is only 2 things different for each FOR loop, the number of steps (3), and the direction 'E'. How about make this even shorter, use a Method? (Programmers are lazy...it's a good thing!)
- You'll need to explain the `def` keyword, how to name the function, and what are parameters...and the `return` keyword.

```
def build_moves(moves, direction, count):
    for i in range(0, count):
        moves.append(direction)
    return moves
```

- Now that the function is created, we can call it, and now our code is very short. Is it easier to read?

```
moves = []
moves = build_moves(moves, 'E', 3)
moves = build_moves(moves, 'S', 8)
moves = build_moves(moves, 'W', 3)
```

- Depending on time, let the kids use a different maze, and try to solve it.

- Review the Objectives, and summerize what was covered in this lesson.

- Reflection: Lost sheep: Luke 15: 1 - 7
    - How does it feel to be lost?
    - How does it feel if someone is looking for us when we are lost?
    - Jesus tells a parable (short story with a main lesson) about a shepherd leaving 99 sheep, and looking for 1 lost. How precious is each sheep to the shepherd?
    - What is the message Jesus is saying to the religious leaders about why he "welcomes sinners and eats with them"?

Scriptural Reflection:

Here's a good reference for Luke 15:
http://www.magpres.org/sermons/a-parable-universe-luke-15

The Parable of the Lost Coin (Luke 15)

15 Now the tax collectors and sinners were all drawing near to hear him. **2** And the Pharisees and the scribes murmured, saying, "This man receives sinners and eats with them."

8 "Or what woman, having ten silver coins,[a] if she loses one coin, does not light a lamp and sweep the house and seek diligently until she finds it? **9** And when she has found it, she calls together her friends and neighbors, saying, 'Rejoice with me, for I have found the coin which I had lost.' **10** Just so, I tell you, there is joy before the angels of God over one sinner who repents."

- Why is the women looking so diligently for the lost coin? Why not just be happy with the 9 she has?
- How would you describe her effort in finding the coin?
- What is her reaction after finding the coin?
- What is the message Jesus is saying to the religious leaders about why he "welcomes sinners and eats with them"?

	(unknown artist)
	Jim Hasse Jesuit priest

- Closing Prayer

Week 3:

Objectives:

5) VS Code:
    a) Debugging?
6) Python code:
    a) IF statements
    b) Comments in our code: `#`
    c) Random generator
    d) Variables: Remember something, some value (in memory).

Opening Prayer

Let's review a bit about last week:

- List of moves
- FOR statement
- Creating a Function.

We are now going to review this present code, and begin to change it. We are going to begin to make our robot move about the maze randomly, and see if it can find the exit, by itself.

Let's look at this:

- Remember, `def` means this is a function, defining a function. This function is called, `callback`.
- First, thing it does is sleep for half a second.
- TODO: Not sure what `global moves` means...need to ask Mark.
- IF statement:
    - Key idea in programming: Make a decision on a condition/situation.
    - In this case, if the program is out of moves (moves has no more steps)
        - Wait (`sleep`) for 5 seconds.
        - Move = "Q" for quit. (end the program)
    - ELSE do the move instructed to do next: `move = moves.pop(0)`
- `The last IF statement does 2 things:`
    - Calls the `callback` method.
    - And then runs that one move.

```
def callback(position):
    time.sleep(0.5)
    global moves
    if len(moves) == 0:
        time.sleep(5)
```

```
        move = 'Q'
    else:
        move = moves.pop(0)
    return move

if __name__ == '__main__':
    app = Runner(callback)
    app.run()
```

Let's create the random movement:

- Out of the 4 directions, randomly pick one of them.
- Explain the range.
- What is an integer?
- What will rolling bit a 4 side die?
- (Do we need a backup function to simplify it?) - `random.choice(moves)`
- Zero-based counting

```
def callback(position):
    time.sleep(0.5)
    moves = []
    for move in ['N', 'S', 'E', 'W']:
        moves.append(move)
    # randomly pick one of the valid moves we selected above
    move = moves[randint(0, 3]
    return move
```

Run and try it out...it may crash or get stuck...why? Well, robot is hitting a wall. Let's fix that.

```
def callback(position):
    time.sleep(0.5)
    moves = []
    for move in ['N', 'S', 'E', 'W']:
        if position.valid(move):
            # move is valid, add it to the list of possibilities
            moves.append(move)
    # randomly pick one of the valid moves we selected above
    move = moves[randint(0, 3]
    return move
```

Let's try it now. Does it work? Well, now sometimes all directions are not valid, but we are still picking from 4 moves: 0, 1, 2 & 3. We need to only pick from valid choices. Let's fix that.

```
def callback(position):
```

```
    time.sleep(0.5)
    moves = []
    for move in ['N', 'S', 'E', 'W']:
        if position.valid(move):
            # move is valid, add it to the list of possibilities
            moves.append(move)
    # randomly pick one of the valid moves we selected above
    move = moves[randint(0, len(moves)-1)]
    return move
```

Now let's try it. How is it going? It's working, but what's the main problem know. Sometimes, the robot is going just back and forth. Let's try to eliminate that….

BREAK: Let's take a mental break, walk around, stretch, close your eyes….

Note: Need to explain what `global last_move` means...need to ask Mark.

- Create a variable to remember the last move.
- Save the last move at the end of the function.
- We need to know what the opposite (inverse) of our move. Let's create some relationship between opposites moves.
- Now, let's create an IF statement to check if the move we are about to take, is the opposite of the last move. If so, don't allow this move to be a candidate for a next move. Just skip it.

- Add more nest IF statements: convert inverses to IF statement

```
last_move = None
inverses = {
    'N': 'S',
    'S': 'N',
    'E': 'W',
    'W': 'E'
    }

def callback(position):
    global last_move

    time.sleep(0.5)
    moves = []
    for move in ['N', 'S', 'E', 'W']:
        if last_move and move == inverses[last_move]:
            # skip
```

```
            continue
        if position.valid(move):
            # move is valid, add it to the list of possibilities
            moves.append(move)
    # randomly pick one of the valid moves we selected above
    move = moves[randint(0, len(moves)-1)]
    # and populate last_move for next time
    last_move = move
    return move
```

OK, did that work. Is it working better? Let's watch our robot now…

Looks like sometimes, our robot is completely stuck. If there are no options, it's at a dead end, it needs to go in the reverse direction. Let's fix that.

- So if there are no moves left, go in the opposite direction of the last move.

```
last_move = None
inverses = {
    'N': 'S',
    'S': 'N',
    'E': 'W',
    'W': 'E'
    }

def callback(position):
    global last_move

    time.sleep(0.5)
    moves = []
    for move in ['N', 'S', 'E', 'W']:
        if last_move and move == inverses[last_move]:
            # skip
            continue
        if position.valid(move):
            # move is valid, add it to the list of possibilities
            moves.append(move)
    # randomly pick one of the valid moves we selected above
    move = moves[randint(0, len(moves)-1)]
    # and populate last_move for next time
    last_move = move
    return move
```

Now, let's watch the robot. How is it doing? Much better...sort of. Now, sometimes, it actually at the exit...and it doesn't even know it. If it's at the exit, stop the program. We solved the maze. Let's add that last step.

```
last_move = None
inverses = {
    'N': 'S',
    'S': 'N',
    'E': 'W',
    'W': 'E'
    }

def callback(position):
    global last_move

    time.sleep(0.5)
    moves = []
    for move in ['N', 'S', 'E', 'W']:
        # short-circuit if we're next to the exit
        if position.is_exit(move):
            # yay!
            return move
        if last_move and move == inverses[last_move]:
            # skip
            continue
        if position.valid(move):
            # move is valid, add it to the list of possibilities
            moves.append(move)
    # randomly pick one of the valid moves we selected above
    move = moves[randint(0, len(moves)-1)]
    # and populate last_move for next time
    last_move = move
    return move

Now, let's watch the robot.  Let's see if it can find the exit.
```

How about let's try to solve another maze. Let's use some batch commands and use different arguments in our maze program.

1) Let's use a different maze. Run the code.
    a) TODO: Let's get a good example from Mark. The command argument is this:

```
--seed
```

2) Let's use a different size of a maze. Run the code.
    a) TODO: Let's get a good example from Mark. The command argument is probably one of these, or both:

```
--complexity

--density
```

From Mark's documentation:

Command line flags:

- --height sets the height of the maze, defaults to 21
- --width sets the width, defaluts to 35
- --hide hides the exit until the bot is adjacent to it
- --seed provides a random seed, useful for retrying a maze after tweaking your algorithm.
- --complexity and --density tweak the internals of the maze generator.

- Review the Objectives, and summerize what was covered in this lesson.

Scriptural Reflection:

Scriptural Reflection:

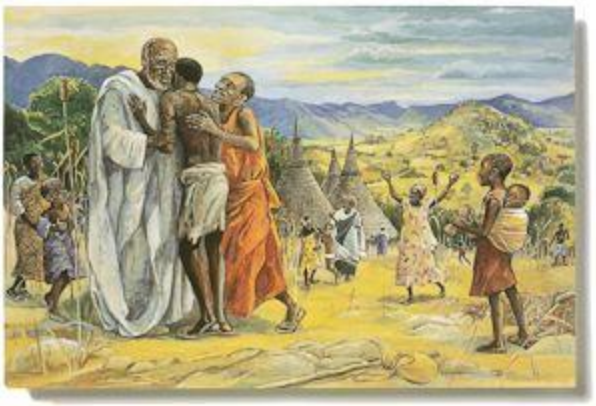
The Parable of the 2 sons - Younger, Prodigal Son (Luke 15: 11-24)

15 **1** Now the tax collectors and sinners were all drawing near to hear him. **2** And the Pharisees and the scribes murmured, saying, "This man receives sinners and eats with them."....

11 Then Jesus[a] said, "There was a man who had two sons. **12** The younger of them said to his father, 'Father, give me the share of the property that will belong to me.' So he divided his property between them. **13** A few days later the younger son gathered all he had and traveled to a distant country, and there he squandered his property in dissolute living. **14** When he had spent everything, a severe famine took place throughout that country, and he began to be in need. **15** So he went and hired himself out to one of the

citizens of that country, who sent him to his fields to feed the pigs. **16** He would gladly have filled himself with the pods that the pigs were eating; and no one gave him anything. **17** But when he came to himself he said, 'How many of my father's hired hands have bread enough and to spare, but here I am dying of hunger! **18** I will get up and go to my father, and I will say to him, "Father, I have sinned against heaven and before you; **19** I am no longer worthy to be called your son; treat me like one of your hired hands."' **20** So he set off and went to his father. But while he was still far off, his father saw him and was filled with compassion; he ran and put his arms around him and kissed him. **21** Then the son said to him, 'Father, I have sinned against heaven and before you; I am no longer worthy to be called your son.'[c] **22** But the father said to his slaves, 'Quickly, bring out a robe—the best one—and put it on him; put a ring on his finger and sandals on his feet. **23** And get the fatted calf and kill it, and let us eat and celebrate; **24** for this son of mine was dead and is alive again; he was lost and is found!' And they began to celebrate.

- Reflection: Younger Son: Luke 15: 11-24
    - How is the younger son treating his father?
    - What made the younger son come to his senses?
    - What is surprising about the father's reaction?
    - What is the message Jesus is saying to the religious leaders about why he "welcomes sinners and eats with them"?



- Closing Prayer

Week 4:

Objectives:

7) Review Randomness and seeds
8) Begin either:
    a) Right-hand rule
    b) Randomness w/ guards
    c) Randomness - knowing the exit position

Opening Prayer

Open VS Code:

Let's open VS Code and review our randomness code. Let's run it using Bash.

Now, let's review what Debugging is:

- Allows programmers to actively step thru each line of code.
- See the values of the variables while you step thru the code.

Let's configure VS Code to allow this to occur

(We need to fill in what these configurations are)

- One of these is to use the Bash arguments in VS Code
- Let's set a breakpoint: This is where want the code to stop.
- Let's "Step Over" each line...notice we can see that value of the variables.

Let's try a different maze, run your random code, and even step thru the code using debugging.

(Allow students practice their code.)

Now, let's take a break and review Randomness:

- Let's go back to VS Code. Let's run some code in the Immediate window.
    - Use the Seed, now run a Random function:

```
randint(0, 10)
```

The result is: `7`

Run the Random function again:

```
randint(0, 10)
```

The result is: `10`

Run the Random function again:

```
randint(0, 10)
```

The result is: `10`

Repeat this again. We'll find the results is again, 7, 10, 10. Is it randome?

How do we make it more random? Change the seed each time.

- ***Let's take a break:***

- Let's start a new topic:
  - We'll pick either Right-hand rule
  - more Randoness (knowing the exit position)
  - Randonmess with guards

9) Finish either:
   a) Right-hand rule
   b) Randomness w/ guards
10) A little presentation on Object-oriented programming
   a) Position
   b) Direction
   c) Object: Makes the decision if you can move in the requested direction (IsValid)
   d) Possibly letting the students fill in the randomness routine in the guard objects
11) How to continue learning:
   a) CodeWizard
   b) Other on-line sites

Scriptural Reflection:

The Parable of the 2 sons - Older Son (Luke 15: 25-32)

15 **1** Now the tax collectors and sinners were all drawing near to hear him. **2** And the Pharisees and the scribes murmured, saying, “This man receives sinners and eats with them.”....

11 Then Jesus[a] said, “There was a man who had two sons. **12** The younger of them said to his father, ‘Father, give me the share of the property that will belong to me.’ So he divided his property between them. ...

23 And get the fatted calf and kill it, and let us eat and celebrate; **24** for this son of mine was dead and is alive again; he was lost and is found!’ And they began to celebrate.

25 “Now his elder son was in the field; and when he came and approached the house, he heard music and dancing. **26** He called one of the slaves and asked what was going on. **27** He replied, ‘Your brother has come, and your father has killed the fatted calf, because he has got him back safe and sound.’ **28** Then he became angry and refused to go in. His father came out and began to plead with him. **29** But he answered his father, ‘Listen! For all these years I have been working like a slave for you, and I have never disobeyed your command; yet you have never given me even a young goat so that I might celebrate with my friends. **30** But when this son of yours came back, who has devoured your property with prostitutes, you killed the fatted calf for him!’ **31** Then the father[d] said to him, ‘Son, you are always with me, and all that is mine is yours. **32** But we had to celebrate and rejoice, because this brother of yours was dead and has come to life; he was lost and has been found.’”

- Reflection: Older Son: Luke 15: 25-32
    - What is making the elder son angry? Can you relate to his reaction?
    - How has the elder son also not known about the father?
    - What is remarkable about the father?
    - What is the message Jesus is saying to the religious leaders about why he “welcomes sinners and eats with them”?

- Closing Prayer