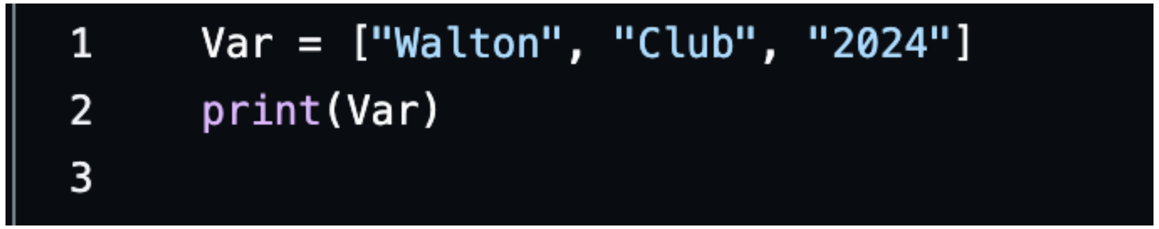
Loops, Arrays and Dictionaries

Week 3

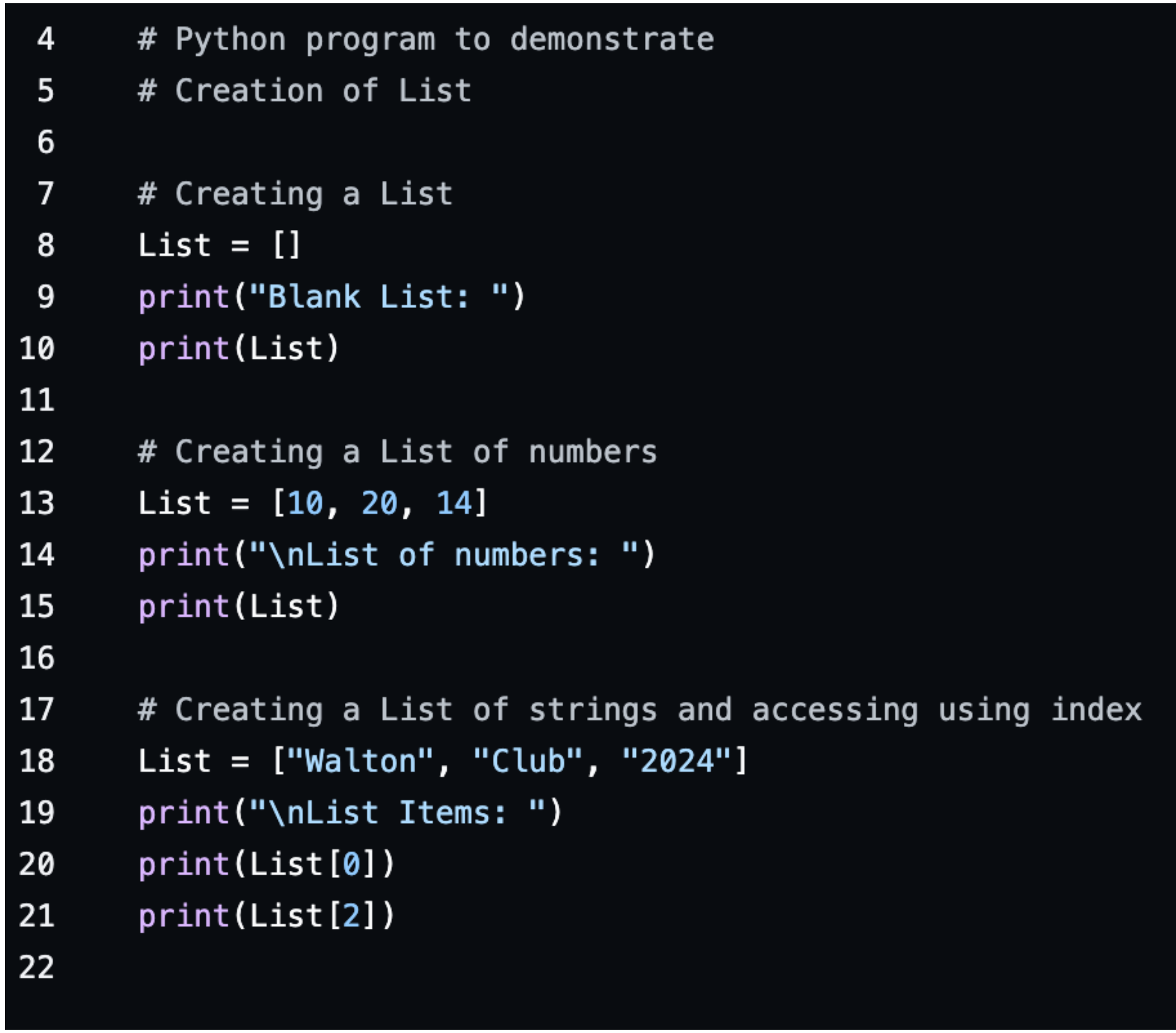
Trinity Walton Club

Lists

A list is a collection of things in python

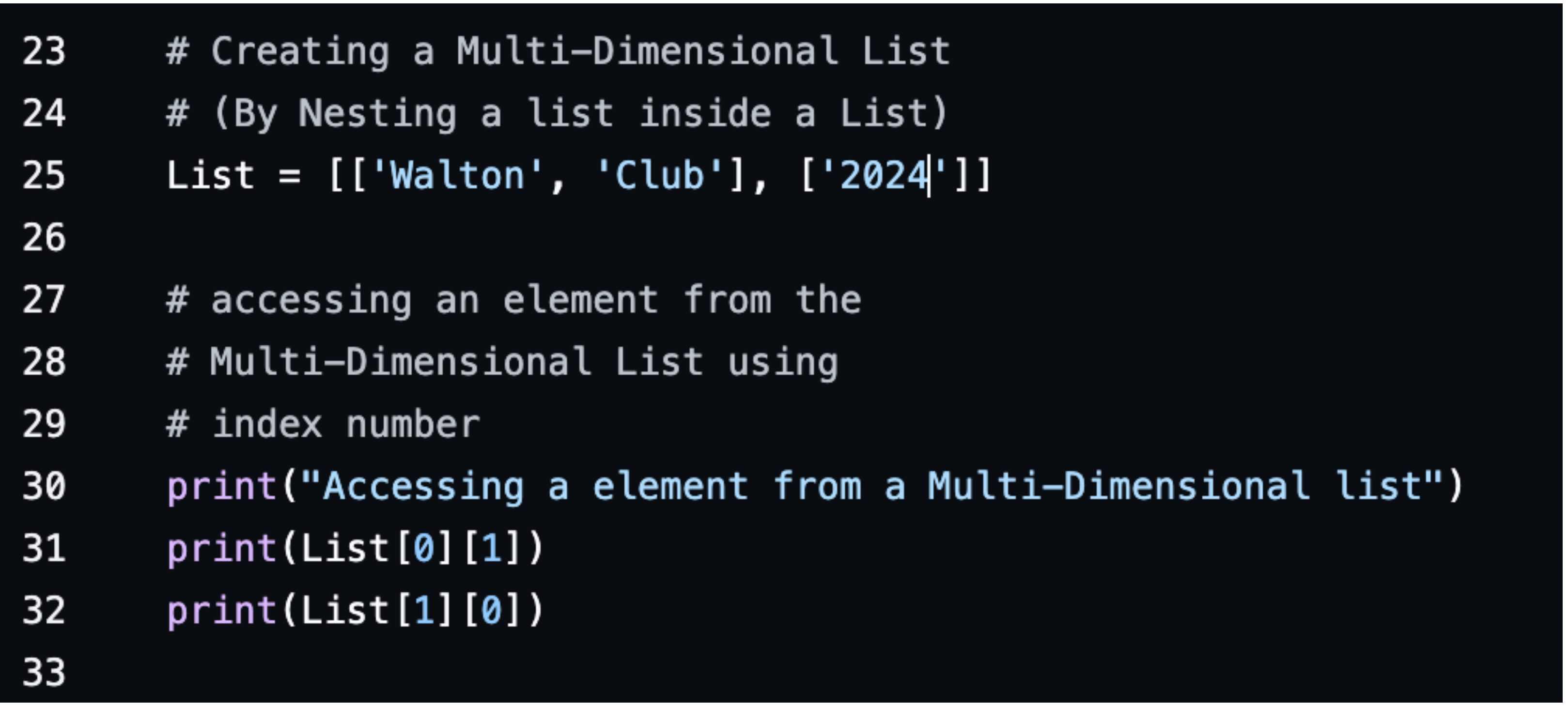


```python
Var = ["Walton", "Club", "2024"]
print(Var)

```

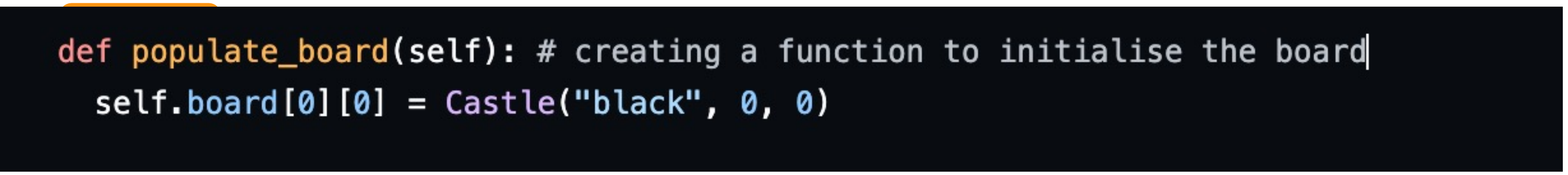


```python
# Python program to demonstrate
# Creation of List

# Creating a List
List = []
print("Blank List: ")
print(List)

# Creating a List of numbers
List = [10, 20, 14]
print("\nList of numbers: ")
print(List)

# Creating a List of strings and accessing using index
List = ["Walton", "Club", "2024"]
print("\nList Items: ")
print(List[0])
print(List[2])

```

2D Lists



```python
# Creating a Multi-Dimensional List
# (By Nesting a list inside a List)
List = [['Walton', 'Club'], ['2024']]

# accessing an element from the
# Multi-Dimensional List using
# index number
print("Accessing a element from a Multi-Dimensional list")
print(List[0][1])
print(List[1][0])

```



```python
def populate_board(self): # creating a function to initialise the board
  self.board[0][0] = Castle("black", 0, 0)
```

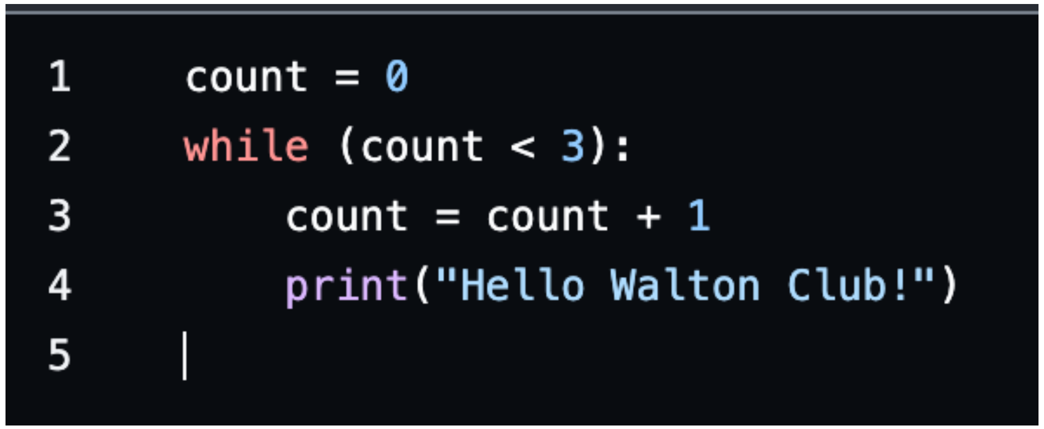
You can also create 2D lists in Python

This can be used to create a 2D array of the chess board pieces in python for printing on the screen.

You can also use the 2D array to access the chess board pieces.

You can initialize the board as the starting positions of the chess pieces, and change the array when moving the chess pieces

While Loops

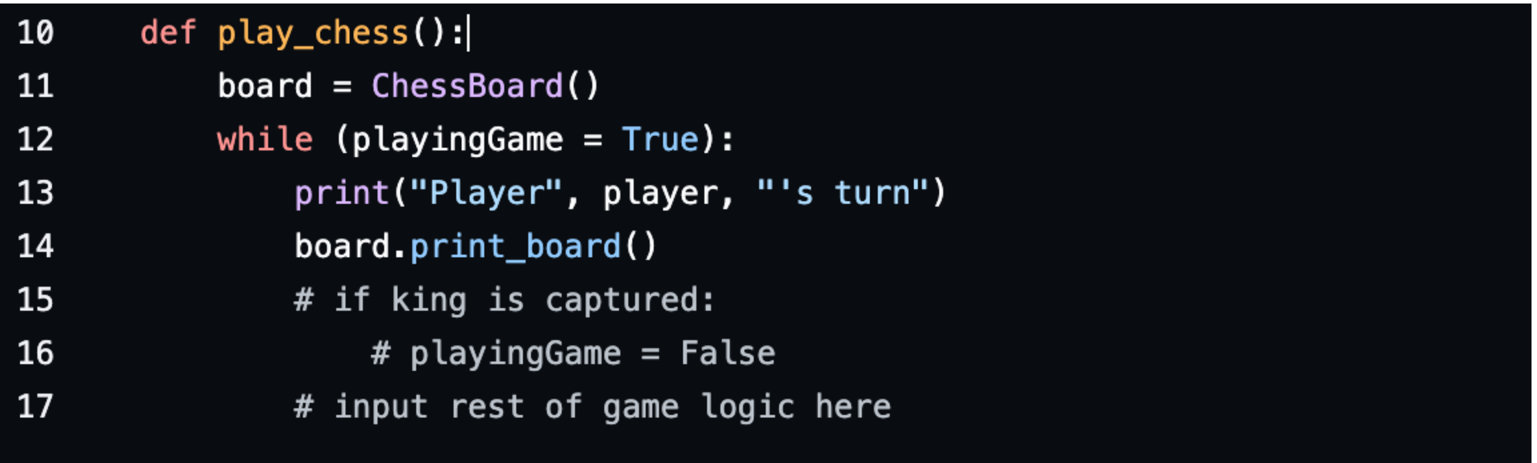


```python
count = 0
while (count < 3):
    count = count + 1
    print("Hello Walton Club!")

```

In Python, a while loop is used to execute a block of statements repeatedly until a given condition is satisfied.

When the condition becomes false, the line immediately after the loop in the program is executed.

This can be used to create the game loop for the chess game



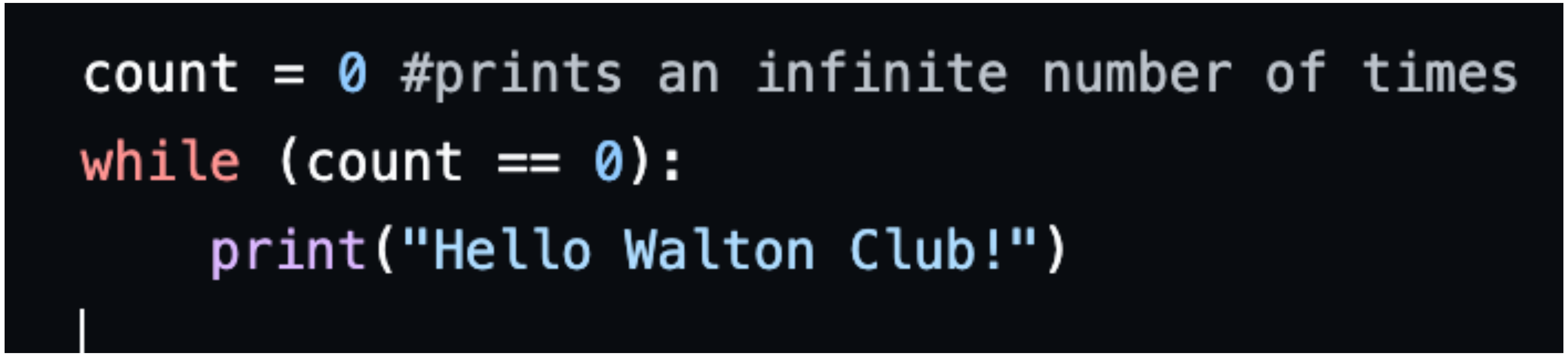
```python
def play_chess():
    board = ChessBoard()
    while (playingGame = True):
        print("Player", player, "'s turn")
        board.print_board()
        # if king is captured:
            # playingGame = False
        # input rest of game logic here
```

Infinite Loops

We can also have loops repeat infinitely, as below.

Make sure your loops have exit conditions (like the addition of 1 to count each time, and checking if count ==3) if you do not want loops to run infinitely

You must create an end condition for the game loop in chess, e.g. when there is a stalemate or a win condition for chess



```
count = 0 #prints an infinite number of times
while (count == 0):
    print("Hello Walton Club!")
```

For Loops

For loops are used for traversing, for example, going through all the letters in a string, all the items in an array, etc.

This is handy for iterating through indexes, e.g. when initializing pawns

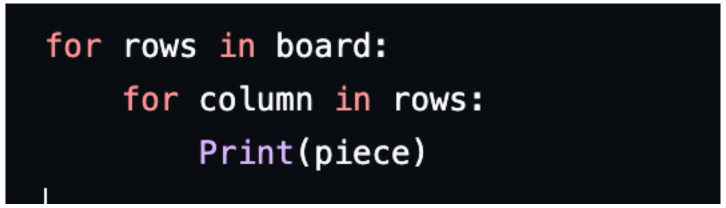
```python
n = 4
for i in range(0, n):
    print(i)

```

```python
def populate_board(self):
    # Populate pawns
    for i in range(8):
        self.board[1][i] = Pawn("black", i, 1)
        self.board[6][i] = Pawn("white", i, 6)
```

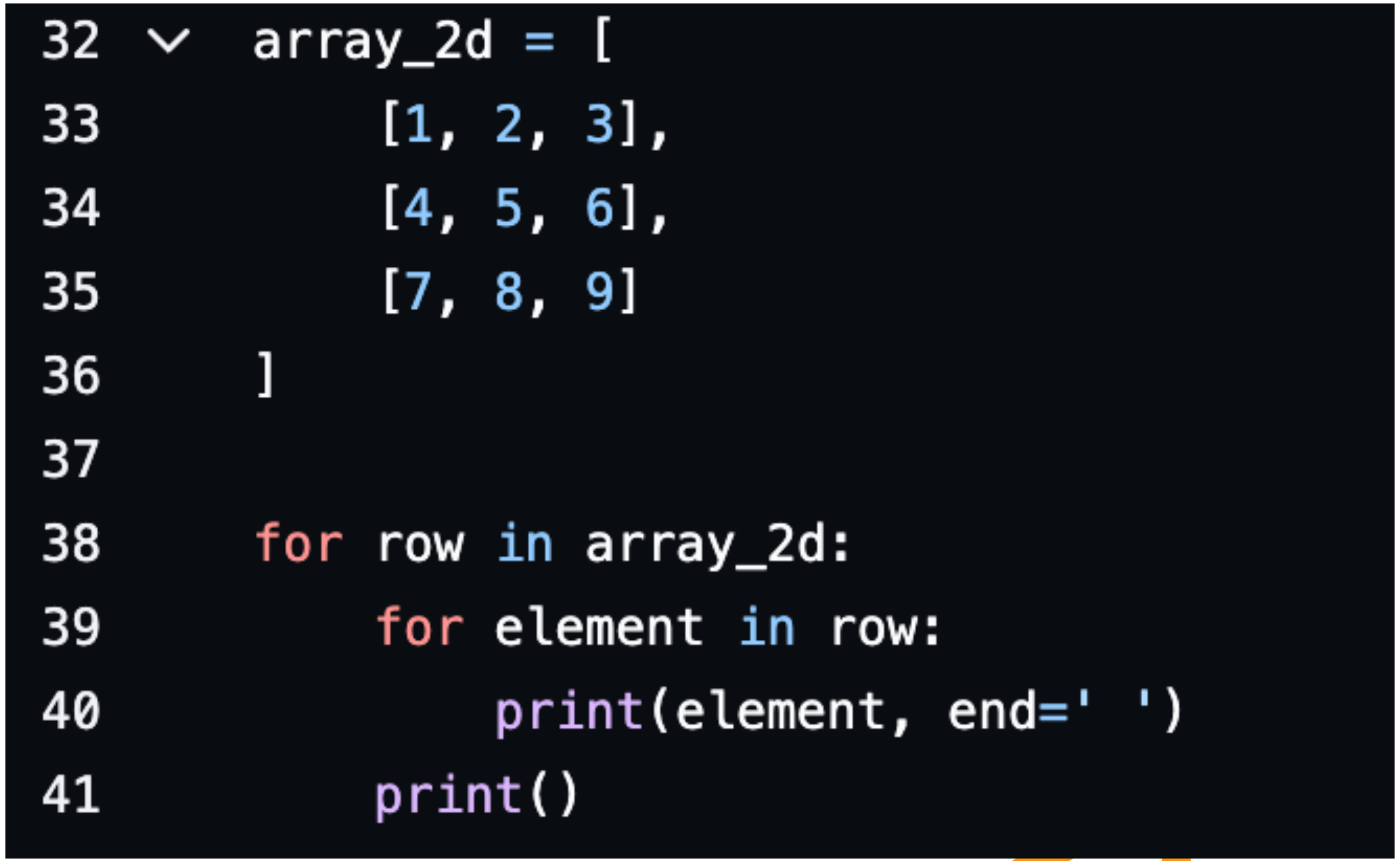
```python
print("List Iteration")
l = ["Walton", "Club", "2024"]
for i in l:
    print(i)
print("\nTuple Iteration")
t = ("Walton", "Club", "2024")
for i in t:
    print(i)
print("\nString Iteration")
s = "Walton"
for i in s:
    print(i)
print("\nDictionary Iteration")
d = dict()
d['xyz'] = 123
d['abc'] = 345
for i in d:
    print("%s  %d" % (i, d[i]))
print("\nSet Iteration")
set1 = {1, 2, 3, 4, 5, 6}
for i in set1:
    print(i),
```

Nested loops

- In python, we are able to have loops within loops
- This is especially handy for accessing the elements in a 2D list, for example when printing out the 2D board with its corresponding pieces
- e.g. the pseudocode:



```
for rows in board:
    for column in rows:
        Print(piece)
```

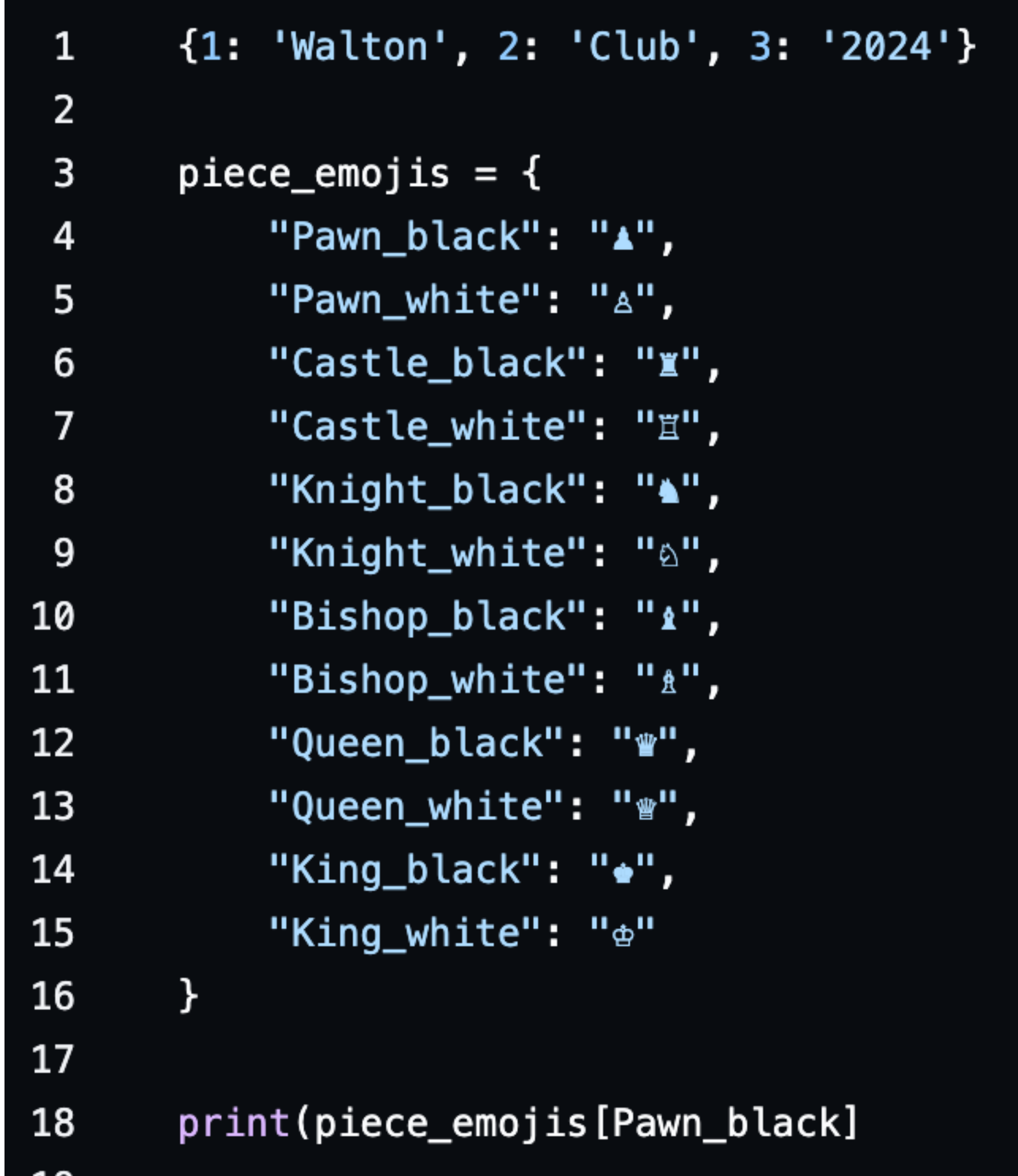


```
array_2d = [
        [1, 2, 3],
        [4, 5, 6],
        [7, 8, 9]
    ]

for row in array_2d:
    for element in row:
        print(element, end=' ')
    print()
```

Dictionaries

A python dictionary is a data structure that stores values in `key : value` pairs

This can also be used for storing the chess piece characters, along with their letter form



```
{1: 'Walton', 2: 'Club', 3: '2024'}

piece_emojis = {
        "Pawn_black": "♟",
        "Pawn_white": "♙",
        "Castle_black": "♜",
        "Castle_white": "♖",
        "Knight_black": "♞",
        "Knight_white": "♘",
        "Bishop_black": "♝",
        "Bishop_white": "♗",
        "Queen_black": "♛",
        "Queen_white": "♕",
        "King_black": "♚",
        "King_white": "♔"
}

print(piece_emojis[Pawn_black]
```

Exercise

- Create a replit account and create a new team project in pygame:
  - https://replit.com/new/pygame
- In your teams chosen last week, complete the task at the following link:
  - https://github.com/b00rg/walton-programming/blob/main/Task_Brief.md
- You may also use the code available here as a starting point:
  - https://github.com/b00rg/walton-programming/blob/main/ChessGame.py
- As before, sample code and slides are available at:
  - https://github.com/b00rg/walton-programming

Sources

- https://www.geeksforgeeks.org/loops-in-python/
- https://www.geeksforgeeks.org/python-lists/
- https://www.geeksforgeeks.org/python-dictionary/
- https://www.codingal.com/coding-for-kids/blog/building-a-chess-game-in-python/