LOOPS

WHILE LOOP

```
i=3
while i != 0:
    print("Meow")
    i -= 1
```

List- used to store multiple items

FOR LOOP

```
#range starts at 0
for i in range(3):
    print("Meow")
```

BOOLEAN LOOP

```
while True:
    n = int(input("Enter number"))
    if n % 2 == 0:
        continue
    else:
        print("Ha")
        break
 #takes n input and implimantes it into for loop
for _ in range(n):
    print("meow")
```

COUNTINUE- Keeps loop going
BREAK-Stops loop
LOOP EXERSISE 1

```
#using loops in mutiple classes
def main():
    meow(3)

def meow(n):
   while n != 0:
       print("meow")
       n-=1
main()
```

ITERATION WITH LIST

```
food = ["apple","pie","cake"]
print(food[0])

for foods in food:
    print(foods)
```

RANGE

```
student=["Jake", "Sam", "Elliot","Paige"]
#use range to set list as boundary, use it to print out list contents
#use - to set starting point or list length
for i in range(len(student)):
    print(i+1, student[i])
```

USE LENGTH OF LIST AS VARIABIE

```
i=0
student=["Jake", "Sam", "Elliot","Paige"]
#use only len if using len of value
while i <= len(student):
    print(student)
    i+=1
```

DICT

```
student={"Jake":"Male",
    "Lisa":"Female",
    "Maria":"Female",
    "Miek":"Male"}
    #sep = chose waht seperates words
for stu in student:
    print(stu, student[stu],sep=", ")
```

DICT PT 2

```
student = [
    {"name":"josh", "Color":"Blue", "Animal":"Rabbit"},
    {"name":"Carl", "Color":"Red", "Animal":"Cow"},
    {"name":"Mary", "Color":"Green", "Animal":"Frog"},
    {"name":"Katie", "Color":"Violet", "Animal":None},
    ]

for stu in student:
    print(stu["name"], stu["Animal"],sep=", ")
```

NESTED LOOPS

```
def main():
    print_col(3)

def print_col(h):
    for i in range(h):
        print("#")

main()
```

LOOP EXERSISE

```
def main():
    print_row(4)

def print_row(w):
    print("@" * w)

main()
```

EXERSISE 2

```
def main():
    print_square(4)

def print_square(w):
    #coloumns loop
    for i in range(w):
        #rows loop
        for j in range(w):
            #end make sure no break after symbol printing
            print("#", end=" ")
        #print new line
        print()
        w-=1
Main()
```

EXERSISE

```
def main():
    print_stair(1,4)

def print_stair(r,c):
    #print 4 columns
    for i in range(c):
        #each row has one
        for j in range(r):
            #end make sure no break after symbol printing
            print("#",end="")

        print()
        r+=1
main()
```

EXERSISE

```
def main():
    r=int(input("Enter the Starting Point:"))
    c=int(input("Enter the nuber of Rows:"))
    print_stair(r,c)
def print_stair(r,c):
    #print 4 columns
    for i in range(c):
        #each row has one
        for j in range(r):
            #end make sure no break after symbol printing
            print("#",end="")
        print()
        r+=1
main()
```

CAMELCASE

```
# TESTS
# convert input from camel to snake case
# name > name
# firstName > first_name
# preferrdFirstName > preferred_first_name

text = input("camelCase: ")
```

```
# loop word for letters
for n in text:
    # check for uppercase and replace with _letter
    if n.isupper():
        newText = "_" + n.lower()
        # replace letter with updated _letter
        n = newText
    #print!
    print(n, end="")
```

COKE

```
# TESTS/FLOW
# - 25, 10, 5 allowed
# - calc how much is owed to reach 50
# - reprompt for only positive integers

print("Amount due: 50")

# #starting amount
changeOwed = 50


while changeOwed > 0:
    coins = int(input("Insert coins: "))

    #reprompt for positive integer
    if coins < 0:
        print("Amount due: 50")

    # valid inputs, subtract coins entered from total change (50)
    if coins == 25 or coins == 10 or coins == 5:
        changeOwed -= coins
        # if change goes below 0, break loop, print coins
        if changeOwed <= 0:
            break
            print(0)
        print("Change owed: " + str(changeOwed))
    else:
        print(50)

#return absolute value returned
print("Change owed: " + str(abs(changeOwed)))
```

VOWELS

```
# TEST CASES
# - Twitter >> Twttr
# - What's your name? >> Wht's yr nm?
# - CS50 >> CS50
```

```
text = input("Input: ")

vowels = ['a', 'e', 'i', 'o', 'u','A','E','I','O','U']
newText = ""

for i in range(len(text)):
    if text[i] not in vowels:
        newText += text[i]

text = newText
print(text)
```

PLATES

```
# TEST CASES
# xx - CS50 >> Valid
# - CS05 >> Invalid
# xx - PI3.14 >> Invalid
# xx - H >> Invalid
# xx - OUTATIME >> Invalid

# REQUIREMENTS:
#  xx - start with 2 letters
#  xx - max 6 chars (letter/num) - min 2 chars
#   - numbers cannot be solely in the middle
#        eg: AAA222 yes, AAA22A no
#  xx - cannot start with 0
#  xx - no periods, spaces or punct
#     - to uppercase?


def main():
    plate = input("Plate: ")

    if is_valid(plate):
        print("Valid")
    else:
        print("Invalid")



def is_valid(s):
    length = len(s)

    # max 6, min 2 chars
    if length >= 2 and length <= 6:
        for letters in s:
            # break if not alpha or num (punct, space, etc case)
            if not s.isalnum():
                break
            #  first 2 char are letters
            if s[0:2].isalpha():
                # middle part of entry
                middle = s[1:-1]
                if middle.isnumeric() and middle.find(0):
                    break

                # if ends with nums, nums cannot start with 0
                # AA022 or CS05 Invalid

                zeroIndex = s.find("0") - 1

                if s[-(zeroIndex)].isdigit():
                    for x in s:
                        if x.isdigit():
                            if x.startswith('0'):
                                return False
                            else:
                                return True

                # true if ends with digit
                if s[-2].isdigit() and s[-1].isalpha():
                    break
                elif s[-2].isdigit():
                    return True
                elif s.isalpha():
                    return True

    else:
        return False

main()
```

NUTRITION

```
#takes data from chosen dictonary
fruits = [
    {"fruit":"apple", "calories":130},
    {"fruit":"avocado", "calories":50},
    {"fruit":"banana", "calories":110},
    {"fruit":"cantaloupe", "calories": 50},
    {"fruit":"grapefruit", "calories": 60},
    {"fruit":"grapes", "calories": 90},
    {"fruit":"honeydew melon", "calories": 50},
    {"fruit":"kiwifruit", "calories": 90},
    {"fruit":"lemon", "calories": 15},
    {"fruit":"lime", "calories": 20},
    {"fruit":"nectarine", "calories": 60},
    {"fruit":"orange", "calories": 80},
    {"fruit":"peach", "calories": 60},
    {"fruit":"pear", "calories": 100},
    {"fruit":"pineapple", "calories": 50},
    {"fruit":"plums", "calories": 70},
    {"fruit":"strawberries", "calories": 50},
    {"fruit":"sweet cherries", "calories": 100},
    {"fruit":"tangerine", "calories": 50},
    {"fruit":"watermelon", "calories": 80}
]

text = input("Item: ")

# prints cals
for x in fruits:
    fruit = x['fruit']
    calories = x['calories']

    if text.lower() == fruit:
        print("Calories: " + str(calories))
```

EXCEPTIONS

ERROR HANDLING: try except

```
try:
    inp=int(input("Enter a number"))
    print(f'Your number is {inp}')
except ValueError:
    print("Not an int")
```

ELSE exception

```
while True:
    try:
        inp=int(input("Enter a number"))
    except ValueError:
        print("Not an int")
    else:
        print(f'Your number is {inp}')
        break
```

GET_INT

```
def main():
   x = get_int()
   print(f'The number is {x}')
def get_int():
    while True:
        try:
            return int(input("Enter a number"))
        except ValueError:
            print("Not an int")
        else:
            return x
main()
```

PASS

```
def main():
   x = get_int()
   print(f'The number is {x}')
def get_int():
    while True:
        try:
            return int(input("Enter a number"))
        except ValueError:
        #pass the loop, does not tell user anything
            pass
        else:
            return x
main()
```

PROMPT

```
def main():
   x = get_int("Enter a number")
   print(f'The number is {x}')
def get_int(prompt):
    while True:
        try:
            return int(input(prompt))
        except ValueError:
        #pass the loop
            pass
        else:
            return x
```

GAS PRICES

```
def main():
    left=get_left()

    if left==0:
        print("No remainder")
    elif left==1:
        print("1")
    elif left==2:
        print("2")
    else:
        print("Too Much")

def get_left():
    while True:
        try:
            text=input("Equation using module")
            num= text.split('%')
            x=int(num[0])
            y=int(num[1])

            if y > x:
                text=input("Equation using module")

            return x%y

        except ValueError:
            pass
        except ZeroDivisionError:
            pass
        except IndexError:
            pass

main()
```

TACO PRICES

#if using classes dosent work just do it by itself

```
food = [
    {"Name":"Baja Taco","Price":4.00},
    {"Name":"Burrito","Price":7.50},
    {"Name":"Nachos","Price":11.00},
    {"Name":"Bowl","Price":8.50}]
x=0
while True:
    try:
        text = input("Enter Food")
        text=text.title()
        for fo in food:
            ite=fo['Name']
            price=fo['Price']
            if text==ite:
                x+=price
    except NameError:
        print("Item not avalable")
    else:
        print(f'The Price is {x}')
```

GROCERRY

```
#create list to insert ittems
groceryList = []
tally = {}

while True:
    try:
        #Takes and captilizes work
        item = input("")
        item = item.upper()
#Combines and sorts list
        groceryList.append(item)
        groceryList.sort()
#count items, can only work on ide
    except EOFError:
        for item in groceryList:
            if item in tally:
                tally[item] += 1
            else:
                tally[item] = 1
        for x in tally:
            print(str(tally[x]) + " " + x)
        break
    else:
        Continue
```

OUTDATED

```
months = [
    "January",
    "February",
    "March",
    "April",
    "May",
    "June",
    "July",
    "August",
    "September",
    "October",
    "November",
    "December"
]

def main():
    formattedDate = validate_date()
    print(formattedDate)

def validate_date():
    date = input("Date: ")

    while True:
        try:
            if (',') in date and ("/") not in date:
                date = date.split(', ')
                year = date[1]
                monthDay = date[0].split(" ")
                day = monthDay[1].zfill(2)
                #connects months to list
                monthIndex = months.index(monthDay[0]) + 1

                #reprompt if days out of bounds
                if int(day) > 31:
                    date = input("Date: ")
                formatted = f"{year}-{monthIndex:02}-{day:02}"
                return formatted

            elif ('/') in date:
                if date.isalnum():
                    date = input("Date: ")

                date = date.split('/')

                # reprompt if spaces
                for x in date:
                    if " " in x:
                        date = input("Date: ")

                month = date[0].zfill(2)
                day = date[1].zfill(2)
                year = date[2]

                # reprompt if out of bounds
                if int(day) > 31 or int(month) > 12:
                    date = input("Date: ")

                formatted = f"{year}-{month}-{day}"
                return formatted

        except ValueError:
            date = input("Date: ")
        else:
            continue

main()
```