For loop:

```
numbers = [1, 2, 3, 4, 5]
for num in numbers:
        print(num)
```

Using range function:

```
for i in range(5):
   print(i)
```

Using enumerate function

```
fruits = ["apple", "banana", "cherry"]
for index, fruit in enumerate(fruits):
   print(index, fruit)
```

Using zip function:

```
names = ["Alice", "Bob", "Charlie"]
ages = [25, 30, 35]
for name, age in zip(names, ages):
   print(name, age)
```

While loop:

```
count = 0
while count < 5:
        print(count)
        count += 1
```

Loop Control Statements:

Break statement:

```
numbers = [1, 2, 3, 4, 5]
for num in numbers:
   if num == 3:
      break
   print(num)
```

Continue Statement:

```
numbers = [1, 2, 3, 4, 5]
for num in numbers:
    if num == 3:
        continue
    print(num)
```

Pass Statement:

```
x = 10
if x > 5:
    pass  # Placeholder, no code executed
else:
    print("x is less than or equal to 5")
```

List comprehension

```
numbers = [1, 2, 3, 4, 5]
squares = [num ** 2 for num in numbers]
```

Generator expression:

```
numbers = [1, 2, 3, 4, 5]
squares_gen = (num ** 2 for num in numbers)
```

Conditional Statement:

If Statement:

```
x = 10
if x > 5:
    print("x is greater than 5")
```

If-else statement:

```
x = 10
if x > 5:
```

```
    print("x is greater than 5")
else:
    print("x is less than or equal to 5")
```

If elif else statement:

```
x = 10
if x > 5:
    print("x is greater than 5")
elif x < 5:
    print("x is less than 5")
else:
    print("x is equal to 5")
```

Nested Statement:

```
x = 10
if x > 5:
    print("x is greater than 5")
elif x < 5:
    print("x is less than 5")
else:
    print("x is equal to 5")
```

Exception Handling:

try-except statement:

```
try:
    # Code that may raise an exception
    result = 10 / 0
except ZeroDivisionError:
    # Handling the specific ZeroDivisionError exception
    print("Error: Division by zero")
except Exception as e:
    # Handling other types of exceptions
    print("Error:", str(e))
```

try-except-else statement

```
try:
```

```
    # Code that may raise an exception
    result = 10 / 5
except ZeroDivisionError:
    # Handling the specific ZeroDivisionError exception
    print("Error: Division by zero")
else:
    # Executed if no exception occurs
    print("Result:", result)
```

try-except-finally statement

```
try:
    # Code that may raise an exception
    file = open("myfile.txt", "r")
    # Perform some operations on the file
except FileNotFoundError:
    # Handling the specific FileNotFoundError exception
    print("Error: File not found")
finally:
    # Always executed, regardless of exceptions
    file.close()
```