CHAPTER 3

DECISION STRUCTURES AND BOOLEAN LOGIC

GROUP 11: JOSEPH HERKNESS AND JOSHUA HERKNESS

IF STATEMENT

- Use the if statement to specify a block of code to be executed if a condition is true.
- Single alternative data structure - has one alternate path

```
if True:
    print('This will execute')
```

IF-ELSE STATEMENT

- Use the else statement to specify a block of code to be executed if a condition is false.
- **Dual alternative data structure** - has two possible paths of execution

```python
if False:
    print('This will NOT execute')
else:
    print('This will execute')
```

```python
# Program 3.4
#
# 1. Prompt the user for a number.
# 2. Display the Roman numeral version of the number.
# 3. If the number is outside the range 0 to 10, display an
#    error message.

def main():

    # Define a dictionary of Roman numerals because we can...
    rn = { 1:'I', 2:'II', 3:'III', 4:'IV', 5:'V',
           6:'VI', 7:'VII', 8:'VIII', 9:'IX', 10:'X'}

    # Ask for the user's input.
    num = int(input('Enter a number between 1 and 10: '))

    # Print either a Roman numeral or an error message.
    if num in rn:
        print(rn[num])
    else:
        print('Sorry, that number is out of range...')

main()
```

RELATIONAL OPERATORS

```
bool_less_than = 10 < 7
bool_greater_than = 3 > 5
bool_less_than_or_equal = 4 <= 2
bool_greater_than_or_equal = 8 >= 4
bool_equal = 9 == 9
bool_not_equal = 6 != 2
```

COMPARING STRINGS

Any relational operator can be used to compare the ascii value of strings.

```python
a = 'Hello'
b = 'World'
if a == b:
    print('The strings are equal')
else:
    print('The strings are NOT equal')
```

NESTED DECISION STRUCTURES

Decision structures can be nested. For example:

```
if 2 + 2 == 4:
    if 42 == 42:
        print('The universe makes sense. 😁')
    else:
        print('The universe doesn\'t makes sense. 🔥')
```

IF-ELIF-ELSE STATEMENT

Use the elif statement to specify a new condition if the previous condition is false.

```python
if False:
    print('This will NOT execute')
elif True:
    print('This will execute')
else:
    print('It will never get here...')
```

```python
# Program 3.3
#
# 1. Enter age
# 2. Given the constraints, print if they are a baby, child, teenager, or adult.

def main():
    age = int(input('Enter your age (in years): '))
    if age <= 1:
        print('Baby 👶')
    elif age > 1 and age < 13:
        print('Child 😣')
    elif age >= 13 and age < 20:
        print('Teenager 😎')
    elif age >=20:
        print('Adult 😴')
    else:
        print('You must not be born yet 😘')
main()
```

LOGICAL OPERATORS

An and statement evaluates to true if both conditions are true.

An or statement evaluates to true if either conditions are true.

A not statement evaluates to the inverse of its condition.

```python
# Program 3.12
#
# 1. Prompt the user to enter the number of packages being ordered
# 2. Display the quantity discount

def main():

    # Get the number of packages
    packages = int(input('Enter the number of packages ordered: '))

    # Display the discount
    if packages >= 10 and packages <= 19:
        print('Discount: 10%')
    elif packages >= 20 and packages <=49:
        print('Discount: 20%')
    elif packages >= 50 and packages <= 99:
        print('Discount: 30%')
    elif packages >= 100:
        print('Discount: 40%')

main()
```

* BOOLEANS*

A boolean is a data type that can either be True or False.

```
a = not False and 7 > 8
b = 'cat' == 'dog'
c = not not not False or False
```

BOOLEANS (ANSWERS)

```
a = False
b = False
c = True
```

BOOLEANS

Boolean variables can be used in decision statements.

```python
a = 'hello' < 'world'
if a:
    print('Hello World!')
```

THANKS!

YOU CAN CLAP NOW... 👏