



PLAN 396

Lecture 9

Dr. Hossen Asiful Mustafa

<https://hossenmustafa.buet.ac.bd>



Variables

○ Naming Variables

- Legal variable names
 - Can only contain letters, numbers, and underscores
 - Can't start with a number
- Good variable names
 - Choose descriptive names
 - Be consistent
 - Follow the traditions of the language
 - Keep the length in check



Variables

- Using Variables

- Once created, it refers to some value
- Use a variable as you would a value

- Example

```
name = "Sarah"
```

```
X = 10
```

```
print name
```

```
Sarah
```



Getting User Input

- Use the Python function
 - `input()` for python 3+
 - `raw_input()` for python 2+

- Example

```
name = input("Please enter your name ")
```

```
name = raw_input("Please enter your name ")
```

- The `input/raw_input` function returns a **string**.
 - Be careful!
 - 10 is not the same as "10"
 - Other functions convert strings to integers and vice versa.



String Methods

- `upper()` Returns the uppercase version of the string
- `lower()` Returns the lowercase version of the string
- `swapcase()` Returns a new string where the case of each letter is switched
- `capitalize()` Returns a new string with the first letter capitalized and the remaining letters are in lowercase
- `title()` Returns a new string with the first letter of each word capitalized and all other letters are in lower case
- `strip()` Returns a new string with leading and trailing white space removed.
- `replace(old, new, [,max])` Returns a new string where occurrences of the string “old” are replaced by “new” up to “max” number of times



Converting Values

- String values can be converted to integers using the `int()` function

- Example

```
x = int( "10" )  
y = input("Enter your age: ")  
y = int( y )  
z = int(input( "Enter your age: " ))
```



Converting Values

- String values can be converted to floating points using the `float()` function

- Example

```
rate = float( "14.5" )
```

```
y = input("Enter interest rate: ")
```

```
y = float( y )
```

```
z = float( input( "Enter interest rate: " ) )
```



Converting Values

- Ints and floats can be converted to string values `str()` function

- Example

```
first4 = 1234
```

```
second4 = 5678
```

```
third4 = 2468
```

```
fourth4 = 3579
```

```
card_number = str( first4 ) + str( second4 ) +  
              str ( third4 ) + str( fourth4 )  
              '1234567824683579'
```


Other Assignment Operators

- Augmented assignment operators

- A combination of assignment and a mathematical operation

● <code>*=</code>	<code>x *= 5</code>	<code>x = x * 5</code>
● <code>/=</code>	<code>x /= 5</code>	<code>x = x / 5</code>
● <code>%=</code>	<code>x %= 5</code>	<code>x = x % 5</code>
● <code>+=</code>	<code>x += 5</code>	<code>x = x + 5</code>
● <code>-=</code>	<code>x -= 5</code>	<code>x = x - 5</code>



Generating Random Numbers

- Programs often must generate random numbers to simulate events that are often based on probability
- You can import the random module into your Python program
- Use the `randrange()` function (method) to generate random numbers in a given range.
 - Not really a “true” random number generator...it is a pseudo-random number generator



Generating Random Numbers

- Use the import statement to include a module
 - Files that contain functions that can be used in any program
 - Import statements are usually at the top of your Python program
- Example
 - `import random`



Generating Random Numbers

- The `randrange()` function will return a random integer in the range `[start..end)`
 - From the value start up to but not including end.
 - Example
 - `anydigit = random.randrange(10)`
 - `[0..10)` -> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
 - `die = random.randrange(6) + 1`
 - `[1..7) = [1..6]` -> 1, 2, 3, 4, 5, 6



Using the if Structure

- Branching is how computers and computer programs make decisions
 - Make a decision to take one path or another

- Example:

```
password="test2"  
if password == "test":  
    print ("You're In!")
```



Comparison Operators

- == equal to
- != not equal to
- > greater than
- < less than
- >= greater than or equal to
- <= less than or equal to



Indentation Blocks

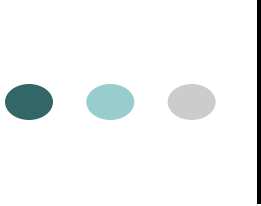
- Code statement blocks in if structures (any control structure) need to be indented
 - tabbed or spaced inside
 - improves readability
 - determines what is the “True” block from other code



Using the if-else Structure

- Sometimes programs need to make a choice
 - if the condition is “True” execute something
 - if the condition is “False” execute something else

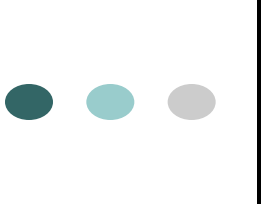
```
if password == "secret":  
    do something  
else:  
    do something else
```

Using the if-elif-else Structure

- If the program needs to choose from more than two possibilities

```
if x == 1:
    y = y+1
else:
    if x== 2:
        y = y+2
    else:
        if x == 3:
            y = y+3
        else:
            y = y-10
```



Using the if-elif-else Structure

- If the program needs to choose from more than two possibilities...use the if-elif-else structure

```
if x == 1:
    y = y+1
elif x == 2:
    y = y+2
elif x == 3:
    y = y+3
else:
    y = y+10
```



Class Assignment

- Write a program named classassignment12.py
- The program should:
 - Take exam number as input from user
 - Show the corresponding grade
 - 80+ = A+
 - 75 – 79 = A
 - 70 – 74 = A-
 - 65 – 69 = B+
 - 60 – 64 = B
 - 55 – 59 = B-
 - 50 – 54 = C+
 - 45 – 49 = C
 - 40 – 44 = D
 - 40- = F