COMPS 101
Principles of Programming

Lecture 6 – Getting user input, converting between types, generating random number
Learning outcomes

- At the end of this lecture, students should be able to:
  - get user input from the keyboard
  - generate a random number
  - convert between types
Recap

- From lecture 5
  - use dot notation when using string functions with string instances
  - use string functions: upper(), lower(), strip(), find(), rfind()
  - use the inbuilt functions: min(), max(), round()
Getting input from the user

- We have already seen how the print() function is used to print to the standard output. We would now like our programs to be able to get input from the user (from the keyboard).

- The `input()` function is used to get information from the user. This function displays the prompt, waits for the user to type their information and, as soon as the user presses the 'Enter' key, the input() function returns the information typed by the user (to the variable on the left of the assignment operator).

```python
user_name = input("Enter name: ")
colour = input("Enter colour: ")
user_word = input("Enter word: ")
print(user_name, "entered", colour, "and the word", user_word)
```

Enter name: **Damir**
Enter colour: **black**
Enter word: **hello**
Damir entered black and the word hello
Getting input from the user

- The `input()` function can be used with no argument (nothing inside the round brackets) in which case there is no prompt printed.

- The `input()` function always **returns a string**. The end of line character is not returned as part of the returned string.

```python
user_number = input("Enter number: ")
user_input = input()

print("The user entered", user_number, "and then", user_input)
```

Enter number: 98
???
#user enters stuff here
The user entered 98 and then ???
Exercise

- Complete the output if the user enters the number 54 when prompted:

```python
user_number = input("Enter number: ")
print(user_number * 2, user_number * 3, user_number * 4)
```

Enter number: 54
The following code causes an error:

```python
1 age = input("Enter age: ")
2 years_to_go = 100 - age
3 print("Big birthday coming up in ", years_to_go, "years!")
```

Enter age: 54
File "LectureCode.py", line 2, in <module>
  years_to_go = 100 - age
TypeError: unsupported operand type(s) for -: 'int' and 'str'

What does this error message mean?
The subtraction operator (-) has no meaning if one of the operands is a string. We want to find a way of converting a string containing just digits into a number.

The `int()` function converts a string containing characters which are digits into the corresponding integer value.

```python
1 age = input("Enter age: ")
2 age = int(age)
3 years_to_go = 100 - age
4 print("Big birthday coming up in ", years_to_go, "years!")
```

Enter age: 54
Big birthday coming up in 46 years!

Note that the code on lines 1 and 2 can be combined:

```python
age = int(input("Enter age: "))
```
Converting between types

Other functions which can be used to convert between types:

- float()
- str()

```python
1 cost = input("Enter cost ")
2 cost = float(cost)
3 final_price = cost * 0.92
4 print("Final price ", final_price, sep="")
```

Enter cost $509.59
Final price $468.8228

String concatenation requires the two operands to be strings:

```python
1 price1 = float(input("Please enter the price of item 1: "))
2 price2 = float(input("Please enter the price of item 2: "))
3 total = price1 + price2
4 message = "The total price of item 1 ($ + str(price1) + " and item 2 ($" + str(price2) + ") is $" + str(total)
5 print(message)
```

Please enter the price of item 1: $7.99
Please enter the price of item 2: $24.99
The total price of item 1 ($7.99) and item 2 ($24.99) is $32.98
Converting between types

- The conversion has to be legal. Below are two illegal attempts to convert:

  ```python
  number = int("12 34")
  File "LectureCode.py", line 1, in <module>
    number = int("12 34")
  ValueError: invalid literal for int() with base 10: '12 34'
  
  number = int("12.34")
  File "LectureCode.py", line 1, in <module>
    number = int("12.34")
  ValueError: invalid literal for int() with base 10: '12.34'
  ```

- Note that converting an string which has no decimal point into a float is fine:

  ```python
  number = float("12")
  print(number)
  12.0
  ```
Getting user input

- Sometimes you are not interested in the input entered by the user, you are just wanting the user to continue when they are ready:

```python
message = " (press Enter to continue) "
question1 = "Think of your favourite person"
question2 = "Think of your least favourite person"

#add the code
colour1 = input("Enter a colour: ")
print()
#add the code
colour2 = input("Enter a colour: ")
print()
print(colour1, "nice,", colour2, "not nice.")
```

Think of your favourite person (press Enter to continue):
Enter a colour: Blue

Think of your least favourite person (press Enter to continue):
Enter a colour: Red

Blue nice, Red not nice.
Random numbers

- Quite often, in our programs, we need to generate random numbers, e.g., for games and simulations.

- The random module contains a function, `randrange()`, which can be used to generate a random number. In order to use this function we need to import the random module into our program (just as we did when we wanted to use the functions of the math class – `math.sin()`, `math.cos()`, …).

Whenever we need to get a random number in a program, the first line of the program will be the following import statement:

```python
import random
```
Random numbers

- The, `randrange()` function requires two values defining the range of the random number to be generated:
  - the first number is the lowest number which can be generated and the second number is one past the biggest number to be generated, e.g., `random. randrange(5, 10)` has an equal chance of generating 5, 6, 7, 8 or 9.

```python
import random

dice1 = random.randrange(1, 7)
dice2 = random.randrange(1, 7)
print("You threw", dice1, "and a", dice2)
```

You threw 4 and a 1
Exercise

- Complete the following program so that it prompts the user for their first name. The program removes a random letter from the first name and prints the resulting name.

```python
import random
first_name = input("Enter name: ")
print(first_name)
```

Enter name: Adriana
Adriaa

Enter name: Eric
ric

Enter name: Jerry
Jery
Random numbers

- The `randrange()` function has an optional step argument:
  ```python
  random.randrange(start, end, step)
  ```

  The default step when the step argument is omitted is 1.
  The step argument dictates which random numbers are generated within the range (the step is the amount between each random number generated), e.g.,

  ```python
  random.randrange(1, 7, 2)  # generates either 1, 3 or 5
  random.randrange(1, 7, 3)  # generates either 1 or 4
  random.randrange(78, 100, 7) # generates either 78, 85, 92, or 99
  random.randrange(-30, 100, 30) # generates either -30, 0, 30, 60, or 90
  ```
Exercise

- Give the smallest and the largest possible random number which can be generated by the following four statements:

```python
import random
print("1.", random.randrange(4, 17, 3))
print("2.", random.randrange(-1, 7, 2))
print("3.", random.randrange(100, 700, 2))
print("4.", random.randrange(50, 100, 10))
```
The backslash character at the end of a line of code in the program indicates that the statement continues onto the next line, e.g.,

```python
user_dice = input("Enter dice throw: ")
comp_dice = random.randrange(1, 7)
message = "Your dice: " + user_dice + ", computer dice: " \ 
+ str(comp_dice)
print(message)
```

Note that the backslash is the last character in the line of code and is not inside a string (i.e., outside the quotes).
**Exercise**

- Complete the following program which prompts the user for a total of 4 dice throws, the program calculates the sum of four random dice throws, and outputs the difference between the user guess and the sum of the dice throwing simulation:

```python

dice_sum = 0

# Enter sum (4 dice):
for _ in range(4):
    print("Enter sum (4 dice):", end=' ')  # Enter your guess
    user_guess = int(input())
    dice_sum += user_guess

print("Your total:", user_guess, "Dice total:", dice_sum)
print("You are out by:", abs(user_guess - dice_sum))
```

<table>
<thead>
<tr>
<th>User Guess</th>
<th>Dice Sum</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>20</td>
<td>8</td>
</tr>
<tr>
<td>15</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>
In a Python program:

- the `input()` function is used to get user input from the keyboard
- a random number can be generated using `random.randint()`
- we can convert between types using `str()`, `int()`, `float()`
Examples of Python features used in this lecture

dice1 = random.randrange(1, 7)
age = random.randrange(66, 99)
even_number = random.randrange(50, 99, 2)
tens = random.randrange(50, 101, 10)

user_input = input("Enter age: ")
age = int(user_input)

cost = input("Enter cost $")
cost = float(cost)

price = 32.45
message = "Final price $" + str(price)