Assignment - 2

1. Short Note on UDF in C.

→ A function is a block of code that performs a specific task

→ C allows user to define functions

→ Function call

→ There are 3 elements for User Defined Functions as follows:

1. Function Declaration
2. Function Definition
3. Function Call

→ 1. Function Declaration

* Syntax

returntype functionName (type1 parameter1, type2 parameter2 ...);

→ Function declaration informs the compiler about the function name, parameters & accept and its return type

→ Function declaration consists

1. Returntype :- It specifies the type of value (int, float, char, double) that function is expected to return to the program which called the function

2. Function Name :- Function name is an identifier and it specifies the name of function.

3. parameter list
→ The parameter list declare the type and number of arguments that the function expects when it is called

4. Terminating Semicolon
→ The terminating semicolon is important as it instructs the compiler that decloration is completed here.

→ 2. UDF Definition
* Syntax

```
returntype function name (type1 parameter1,
                          type2 parameter2 ... )
{
  // function body
}
```

→ Function Body
→ The function body contains declarations and the statements necessary for performing the required task
→ The body enclosed within curly braces {} and consists of three parts
1) local variable declaration variable which are used inside the function body.
2) function statements to perform the task inside the function
3) return statement to return the result evaluated by the function

2. Explain Python variable in detail.

→ Variables: variables are containers for storing data.

→ Rules for Python variable

→ A variable can have a short name or a more descriptive name.

→ A variable name must start with a letter or the underscore character

→ A variable name cannot start with a number

→ A variable name can only contain alpha-numeric character and underscore

→ Variable names are case-sensitive

Valid Variable ✓	Invalid variable ✗
myVar = "John"	2var = "John"
my_var = "John"	v a = "John
myvar = "John"	-va = "John"

→ Creating variable

→ Python has no command for declare variable

→ Variable is created at assign a value.

→ Don't have to declare any type.

→ Example

```
x = 4    # x is int type
z = "cab" # z is string
print(x)
```

→ Many values to Multiple variable.
→ Python allows to assign values to multiple variable in one line

→ Example:

```
x,y,z = "A", "B","C"
print(x)
print(y)
print(z)
```

→ output

```
A
B
C
```

→ ~~Many~~ One values to Multiple variable

→ Ex

`x = y = z = 1`	output
`print(x)`	1
`print(y)`	1
`print(z)`	1

⟹ Global variables
→ Variables that are created outside of a function are known as global variable
→ Global variables can be used by everyone, both inside of functions and outside
→ Example:

```
x = 'awesome'
def fun():
    print("Python is " + x)
```

```
fun()
```

Output:
Python is awesome

→ If you create variable with same name inside a function, this variable will be local, and can only be used inside the function.

⟹ The global keyword

→ Normally, when you create a variable inside a function, that variable is local, and can only be used inside that function.

→ To create a global variable inside a function, you can use the global keyword

→ Example:

```
def fun():
    global x
    x = "fun"

fun()
print(x)
```

→ output:

```
fun
fun
```

3. What is Type conversion in Python? Show with examples.

→ Type conversions and casting

→ If you want to specify the data type of a variable this can be done with casting.

→ Example	Datatype
x = str("Hello")	String
x = int(2)	int
x = float(20.5)	float
x = complex(1j)	complex
x = bool(5)	bool

→ Python Casting

→ There may be times when you want to specify a type on to a variable. This can be done with casting. Python is an object-oriented language and as such it uses classes to define data types, including its primitive types.

→ Casting in python is therefore done using functions-as follows:

→ int() - an integer number from an integer literal, a float literal, or a string literal

→ float() - a float number from an integer literal, a float literal or a string literal

→ str() - a string from a wide variety of

data types including strings, integer literals and float literals.

→ Example int()

Code	Output
x = int(1)	1
y = int(2.8)	2
z = int("3")	3

→ float()

x = float(1)	1.0
y = float(2.8)	2.8
z = float("30")	30.0

→ str()

x = str(1)	"1"
y = str("2.3")	"2.3"
z = str(0.5)	"0.5"

4. What are function with default parameter value and with return value

→ Default parameter value
→ This value will be use when value is empty while calling function.
→ Example

```
def mfun(country = "India"):
    print("I am from " + country)

mfun("USA")
mfun()  # It will use default para value
```

Output
I am from USA
I am from India

→ Function with Return values
→ We are using "return" keyword
→ to let a function return a value, use the return statement

→ Example

```
def fun(x):
    return 2 * x

fun(5)
fun(2)
```

Output
10
4