

Class #01: Setting Up Environment, Variables, Data Types & Type Conversion:

- Installing and configuring the Python environment.
- Understanding basic Python syntax: statements, indentation, and comments.
- Working with variables in Python.
- Exploring different data types in Python.
- Performing type conversions in Python.

মেশিন লার্নিং শিখার ধাপসমূহ:

1. প্রোগ্রামিং ভাষা শিখুন (Python, R):

মেশিন লার্নিং শিখতে প্রথমেই প্রোগ্রামিং ভাষার উপর শক্ত ভিত্তি তৈরি করা জরুরি। বর্তমানে Python এবং R সবচেয়ে জনপ্রিয় ভাষা, Python মেশিন লার্নিং এবং ডেটা সায়েন্সে ব্যাপকভাবে ব্যবহৃত হয়।

2. প্রোগ্রামিং লাইব্রেরি (Built-in Functions) ব্যবহার শিখুন:

Python এর বিভিন্ন লাইব্রেরি যেমন NumPy, Pandas, Matplotlib, Scikit-learn ইত্যাদি সম্পর্কে জানুন এবং এই লাইব্রেরিগুলি ব্যবহার করে বিভিন্ন ডেটা প্রসেসিং ও অ্যানালাইসিস করুন। এদের মাধ্যমে ডেটা ম্যানিপুলেশন, ভিজুয়ালাইজেশন এবং মডেল ট্রেনিং করা যায়।

3. লাইব্রেরি ব্যবহার করে মডেল তৈরি করুন:

মেশিন লার্নিং মডেল তৈরি করতে আপনাকে লাইব্রেরিগুলি ব্যবহার করে বিভিন্ন অ্যালগোরিদম যেমন লিনিয়ার রিগ্রেশন, ক্লাস্টারিং, ডিপ লার্নিং ইত্যাদি প্রয়োগ করতে হবে। Scikit-learn, TensorFlow, Keras, PyTorch এসব লাইব্রেরির মাধ্যমে মডেল ট্রেনিং এবং পরীক্ষার কাজ করবেন।

4. SQL শিখুন (Data Scraping, Data Mining, RDBMS):

মেশিন লার্নিংয়ের ক্ষেত্রে ডেটা সংগ্রহ এবং প্রক্রিয়াকরণের জন্য SQL (Structured Query Language) শিখতে হবে। SQL ব্যবহার করে ডেটাবেস থেকে তথ্য সংগ্রহ, ডেটা স্ক্র্যাপিং, ডেটা মাইনিং এবং RDBMS (Relational Database Management System) সম্পর্কে জানাও অত্যন্ত গুরুত্বপূর্ণ।

5. পরিসংখ্যান শিখুন (Statistics):

মেশিন লার্নিংয়ের মূল ভিত্তি হল পরিসংখ্যান। ডেটা অ্যানালাইসিস এবং মডেল পারফরম্যান্স ইভালুয়েশন করতে পরিসংখ্যানের বিভিন্ন কনসেপ্ট যেমন Probability, Hypothesis Testing, Regression Analysis, A/B Testing ইত্যাদি সম্পর্কে ভালো ধারণা থাকতে হবে।

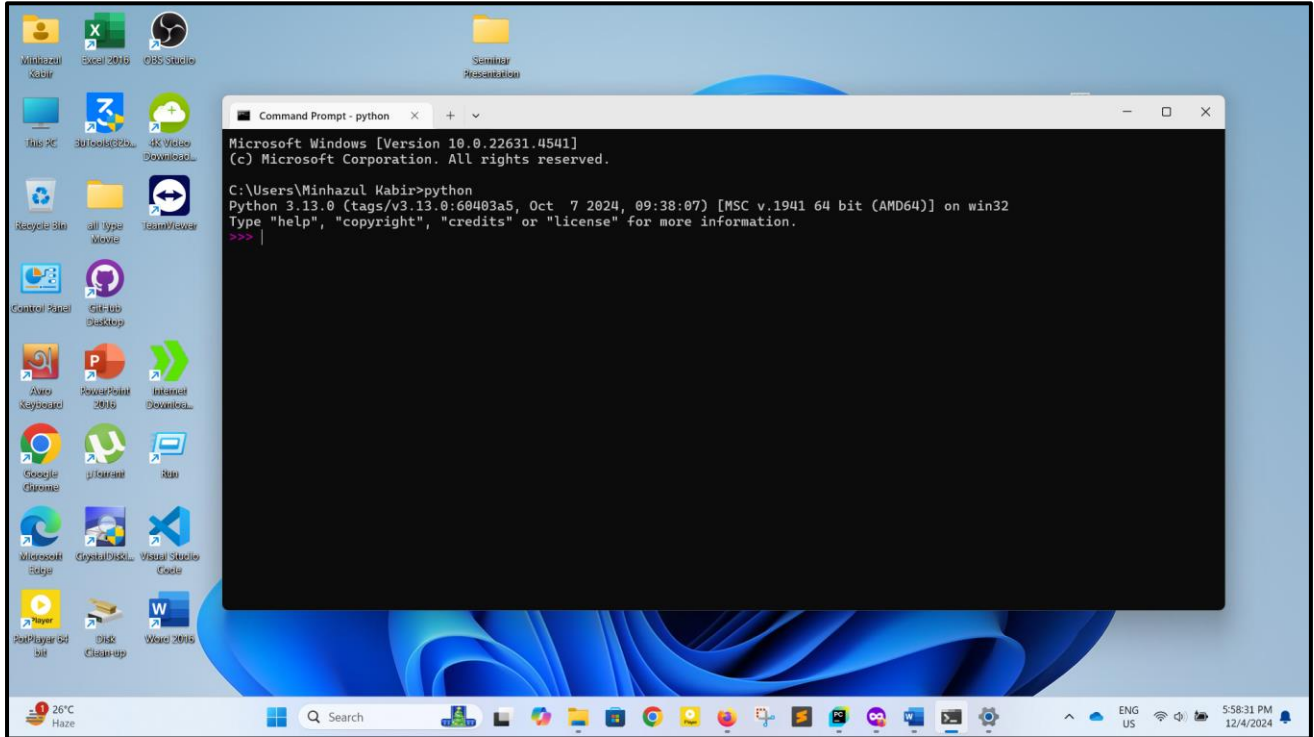
❖ Installing and configuring the Python environment.

Python রান (চালু) করানোর জন্য প্রয়োজন Python এর কম্পাইলার। সেজন্য:

1. Python ডাউনলোড করতে হবে: Python এর অফিসিয়াল ওয়েবসাইট থেকে Python ডাউনলোড করা উচিত। ওয়েবসাইট: <https://www.python.org/downloads/>

মনে রাখবেন: এই ওয়েবসাইট ছাড়া অন্য কোন জায়গা থেকে Python ডাউনলোড করা যায় না।

2. Python ইনস্টল করার পর: কম্পিউটারে Command Prompt (cmd) চালু করে python লিখে Enter চাপলে, নিচের মত কিছু আউটপুট দেখতে পাওয়া যাবে:



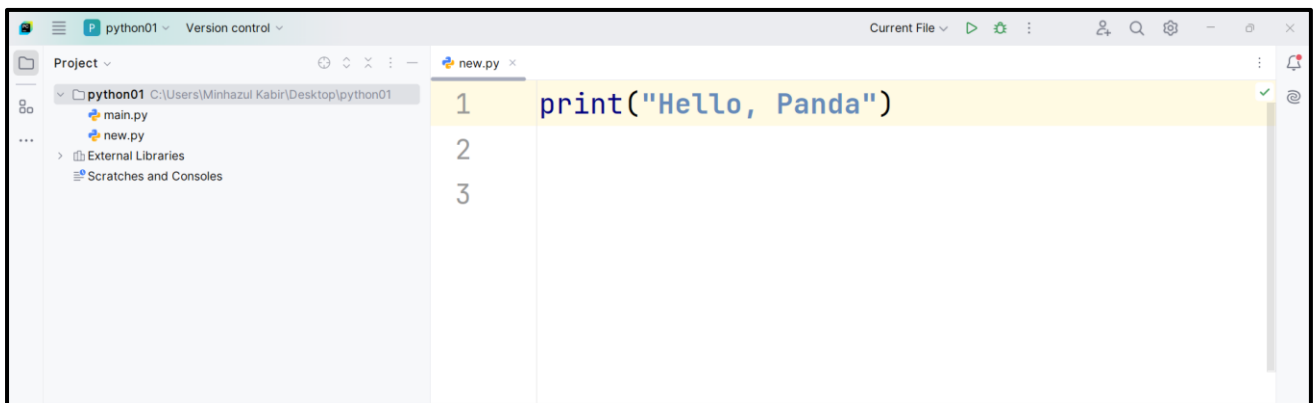
Python কোড লেখার জন্য একটি Text Editor প্রয়োজন। কিছু জনপ্রিয় Text Editor যা Python লিখার জন্য ব্যবহার করা যায়: PyCharm, Visual Studio Code, Jupyter Notebook, Spyder, Notepad, Sublime Text.

PyCharm ডাউনলোড করার জন্য:

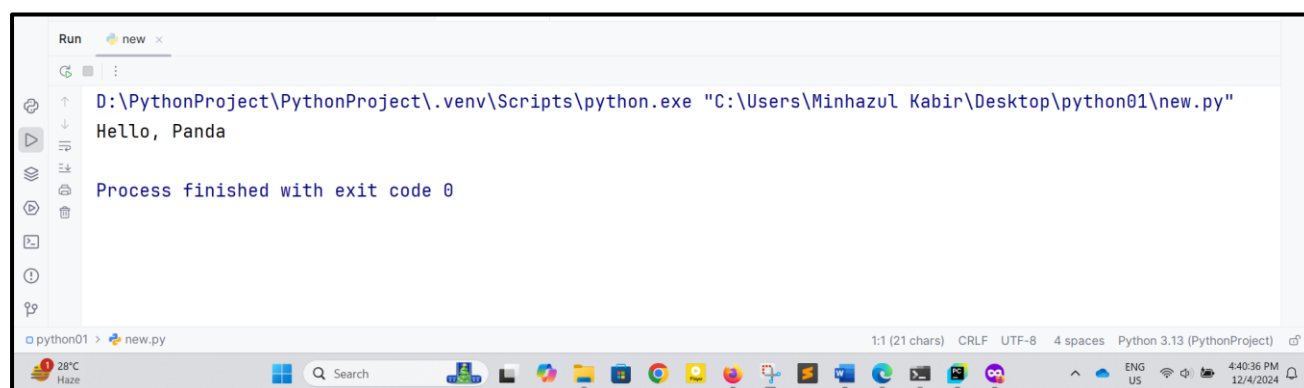
PyCharm হল একটি অত্যন্ত জনপ্রিয় এবং শক্তিশালী Python IDE (Integrated Development Environment)। এটি ডাউনলোড করতে হবে অফিসিয়াল ওয়েবসাইট থেকে:

<https://www.jetbrains.com/pycharm/download/?section=windows>

Developer Part: এটি সেই অংশ যেখানে প্রোগ্রামাররা কোড লেখেন এবং প্রোগ্রাম তৈরি করেন।



User (Output) Part: এটি সেই অংশ যেখানে ব্যবহারকারীরা আউটপুট বা ফলাফল দেখতে পান।



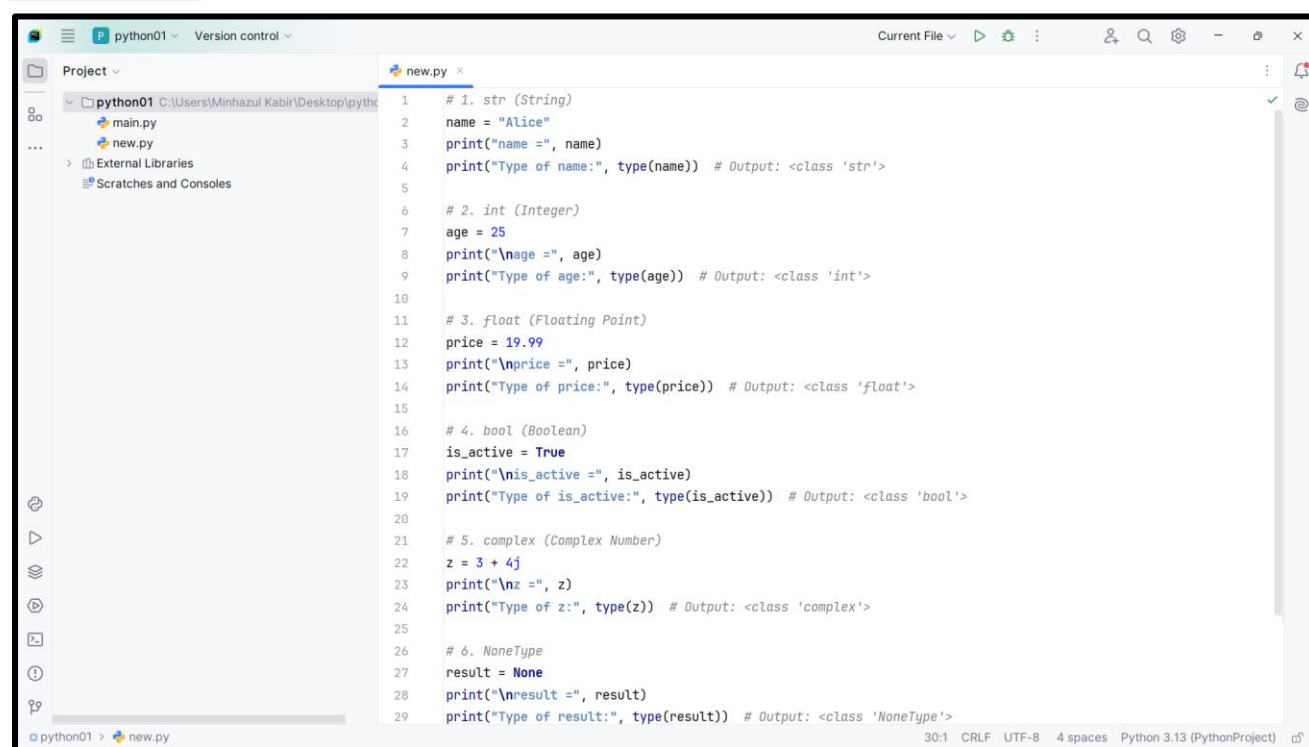
```

Run new x
D:\PythonProject\PythonProject\.venv\Scripts\python.exe "C:\Users\Minhazul Kabir\Desktop\python01\new.py"
Hello, Panda
Process finished with exit code 0
python01 > new.py
1:1 (21 chars) CRLF UTF-8 4 spaces Python 3.13 (PythonProject)

```

❖ Exploring different data types in Python.

Data Type:



```

python01 Version control
Project python01 C:\Users\Minhazul Kabir\Desktop\python01
main.py
new.py
External Libraries
Scratches and Consoles
new.py
1 # 1. str (String)
2 name = "Alice"
3 print("name =", name)
4 print("Type of name:", type(name)) # Output: <class 'str'>
5
6 # 2. int (Integer)
7 age = 25
8 print("\nage =", age)
9 print("Type of age:", type(age)) # Output: <class 'int'>
10
11 # 3. float (Floating Point)
12 price = 19.99
13 print("\nprice =", price)
14 print("Type of price:", type(price)) # Output: <class 'float'>
15
16 # 4. bool (Boolean)
17 is_active = True
18 print("\nis_active =", is_active)
19 print("Type of is_active:", type(is_active)) # Output: <class 'bool'>
20
21 # 5. complex (Complex Number)
22 z = 3 + 4j
23 print("\nz =", z)
24 print("Type of z:", type(z)) # Output: <class 'complex'>
25
26 # 6. NoneType
27 result = None
28 print("\nresult =", result)
29 print("Type of result:", type(result)) # Output: <class 'NoneType'>
python01 > new.py
30:1 CRLF UTF-8 4 spaces Python 3.13 (PythonProject)

```

Python-এ **str**, **int**, **float**, **bool**, **complex** এবং **None** হল মূল ডেটা টাইপ:

1. **str** (String): এটি টেক্সট বা অক্ষরের সিকোয়েন্স। উদাহরণ:

```
name = "Alice"
```

2. **int** (Integer): পূর্ণসংখ্যা, যেমন 5, -3, 100। দশমিক সংখ্যা নয়। উদাহরণ:

```
age = 25
```

3. **float** (Floating Point): দশমিক সংখ্যা, যেমন 3.14, -2.5। উদাহরণ:

```
price = 19.99
```

4. **bool** (Boolean): দুটি মান, True বা False। সাধারণত শর্ত নির্ধারণে ব্যবহৃত হয়। উদাহরণ:

```
is_active = True
```

5. **complex** (Complex Number): জটিল সংখ্যা, যেমন $3+4j$ । উদাহরণ:

```
z = 3 + 4j
```

6. **NoneType**: এটি Python-এ একটি বিশেষ মান, যা "কোনো মান নেই" বা "অর্থহীন" বোঝায়। উদাহরণ:

```
result = None
```

সংক্ষেপে:

- **str**: টেক্সট বা অক্ষরের সিকোয়েন্স।
- **int**: পূর্ণসংখ্যা (ডেসিমাল নয়)।
- **float**: দশমিক সংখ্যা।
- **bool**: বুলিয়ান মান, শুধুমাত্র True বা False।
- **complex**: জটিল সংখ্যা।
- **None**: কোন মান নেই বা এটি একটি বিশেষ টাইপ, যা Python-এ ব্যবহৃত হয় যখন কোন ভেরিয়েবলের মান নেই বা নির্ধারিত নয়।

এই ডেটা টাইপগুলোর মাধ্যমে Python-এ বিভিন্ন ধরনের ডেটা সঞ্চয় এবং পরিচালনা করা সম্ভব।

```

D:\PythonProject\PythonProject\.venv\Scripts\python.exe "C:\Users\Minhazul Kabir\Desktop\python01\new.py"
name = Alice
Type of name: <class 'str'>

age = 25
Type of age: <class 'int'>

price = 19.99
Type of price: <class 'float'>

is_active = True
Type of is_active: <class 'bool'>

z = (3+4j)
Type of z: <class 'complex'>

result = None
Type of result: <class 'NoneType'>

Process finished with exit code 0
  
```

উল্লেখিত ডেটা ধরনের মধ্যে যে ডেটা টাইপগুলো শুধুমাত্র একটি ভেরিয়েবল সংরক্ষণ করে, সেগুলো হল:

1. Numeric Types:

- **int** (ইন্টিজার): একক পূর্ণসংখ্যা মান সংরক্ষণ করে।

- **float** (ফ্লোট): একক দশমিক সংখ্যা সংরক্ষণ করে।
- **complex** (কমপ্লেক্স): একক জটিল সংখ্যা সংরক্ষণ করে।

2. Boolean Type:

- **bool**: একক বুলিয়ান মান (True বা False) সংরক্ষণ করে।

3. None Type:

- **NoneType**: একক None মান সংরক্ষণ করে, যা সাধারণত কোন ভেরিয়েবলের মান না থাকলে ব্যবহার করা হয়।

এগুলো প্রতিটি শুধু একটি মান ধারণ করে, যা একটিই হতে পারে।

বাকি ডেটা টাইপগুলো, যেমন **sequence types** (list, tuple, range), **mapping type** (dict), **set types** (set, frozenset), এবং **binary types** (bytes, bytearray, memoryview) বিভিন্ন উপাদান বা মান ধারণ করতে পারে, তাই তারা একাধিক মান বা উপাদান সংরক্ষণ করে।

Function, Parenthesis, Curly Bracket, Square Bracket:

ফাংশন একটি কোড ব্লক যা নির্দিষ্ট কাজ সম্পাদন করে। প্রথম ব্র্যাকেট () কে **Parenthesis** বলে। কোথাও () পাওয়া গেলে সেটা ফাংশন হয়। দ্বিতীয় ব্র্যাকেট {} কে **Curly Bracket** বলে, যা ব্লক তৈরিতে ব্যবহার হয়। তৃতীয় ব্র্যাকেট [] কে **Square Bracket** বলে, যা তালিকা, ইনডেক্সিং বা স্লাইসিংয়ে ব্যবহৃত হয়।

Python অনুশীলন করার জন্য কিছু গুরুত্বপূর্ণ website:

- <https://www.geeksforgeeks.org/python-programming-language-tutorial/>
- https://www.w3schools.com/python/python_variables.asp

❖ Working with variables in Python.

Variable/ চলকের নামকরণঃ

Python-এ **variable** বা **চলকের নামকরণ** করার জন্য কিছু নির্দিষ্ট নিয়ম রয়েছে। এই নিয়মগুলি অনুসরণ করলে কোডটি পরিষ্কার, সহজবোধ্য এবং মানানসই হয়। নিচে Python এর চলকের নামকরণের মূল নিয়মগুলি দেয়া হলো:

1. চলকের নাম প্রথমে অক্ষর বা আন্ডারস্কোর দিয়ে শুরু করতে হবে:

চলকের নাম কখনোই সংখ্যা দিয়ে শুরু করা যাবে না। এটি অবশ্যই একটি অক্ষর (A-Z বা a-z) বা আন্ডারস্কোর _ দিয়ে শুরু হতে হবে।

- সঠিক: `age, _name, total_volume`
- ভুল: `1st_value, 2nd_number`

2. চলকের নাম শুধুমাত্র অক্ষর, সংখ্যা এবং আন্ডারস্কোর (_) থাকতে পারে:

চলকের নামের মধ্যে অন্য কোন চিহ্ন (যেমন, @, #, &) ব্যবহার করা যাবে না। শুধুমাত্র অক্ষর (A-z), সংখ্যা (0-9) এবং আন্ডারস্কোর _ ব্যবহার করা যাবে।

- সঠিক: `car_name, total_amount, height1`
- ভুল: `car@name, price#value`

3. চলকের নাম কেস-সেনসিটিভ:

Python এ চলকের নাম কেস-সেনসিটিভ। অর্থাৎ, `age`, `Age` এবং `AGE` আলাদা আলাদা চলক হিসেবে বিবেচিত হবে।

- সঠিক: `score, Score, SCORE`
- ভুল: `score` এবং `Score` একে অপরের পরিবর্তে ব্যবহার করা যাবে না।

4. Python কিওয়ার্ড ব্যবহার করা যাবে না:

Python-এর সংরক্ষিত শব্দ (যেমন, `if`, `else`, `True`, `False`, `while`, `class`, `def`, ইত্যাদি) চলকের নাম হিসেবে ব্যবহার করা যাবে না, কারণ এগুলি Python ভাষার অভ্যন্তরীণ কাজের জন্য সংরক্ষিত।

- সঠিক: `total_amount, user_input`
- ভুল: `if`, `else`, `True`, `def`

5. নাম বর্ণনামূলক হওয়া উচিত:

চলকের নাম এমন হওয়া উচিত যাতে তার মান বা উদ্দেশ্য পরিষ্কারভাবে বোঝা যায়। একক অক্ষরের নাম যেমন `x`, `y` ব্যবহার করা উচিত না, যদি না সেটি অস্থায়ী বা নির্দিষ্ট উদ্দেশ্য থাকে।

- সঠিক: `age, user_name, total_price`
- ভুল: `x, y, a`

6. অতিরিক্ত দীর্ঘ নাম এড়িয়ে চলা:

চলকের নাম খুবই দীর্ঘ বা জটিল করা উচিত নয়। তবে, যদি প্রয়োজন হয়, তাহলে বিভিন্ন অংশ আলাদা করতে আন্ডারস্কোর _ ব্যবহার করা যেতে পারে (snake_case পদ্ধতি)।

- সঠিক: `total_price, user_age`
- ভুল: `thetotalpriceofthesystemisveryexpensive`

উপসংহার:

Python-এ চলকের নামকরণ নিয়ম অনুসরণ করলে কোড সহজে পড়া এবং বুঝা যায়, যা ভবিষ্যতে কোড রিভিউ, ডিবাগিং এবং রক্ষণাবেক্ষণকে সহজ করে তোলে।

Assignment Operator হচ্ছে = (সমান) চিহ্নঃ

Python-এ **assignment operator** (=) ব্যবহৃত হয় একটি চলকের মধ্যে মান (value) সংরক্ষণ করার জন্য। এটি চলকের মান নির্ধারণ করে এবং সেই মানটি চলকটির সাথে সম্পর্কিত থাকে। চলকের মান পরিবর্তন করতে assignment operator ব্যবহার করা হয়।

```

1  n = 10 # এখানে 'n' চলককে 10 মান দেওয়া হচ্ছে
2  print(n) # 'n' এর মান 10 প্রিন্ট হবে
3  n = 60 # 'n' এর মান 60 হয়ে যাবে
4  n = 70 # এরপর 'n' এর মান 70 হয়ে যাবে
5  n = 100 # শেষে 'n' এর মান 100 হয়ে যাবে
6  print(n) # 'n' এর সর্বশেষ মান 100 প্রিন্ট হবে
7

```

Run new x

```

D:\PythonProject\PythonProject\.venv\Scripts\python.exe "C:\Users\Minhazul Kabir\Desktop\
10
100

```

Process finished with exit code 0

যেমন আপনার প্রদত্ত কোডটি ব্যাখ্যা করা যাক:

```

n = 10 # এখানে 'n' চলককে 10 মান দেওয়া হচ্ছে
print(n) # 'n' এর মান 10 প্রিন্ট হবে
n = 60 # 'n' এর মান 60 হয়ে যাবে
n = 70 # এরপর 'n' এর মান 70 হয়ে যাবে
n = 100 # শেষে 'n' এর মান 100 হয়ে যাবে
print(n) # 'n' এর সর্বশেষ মান 100 প্রিন্ট হবে

```


ব্যাখ্যা:

1. **n = 10:** এখানে n চলককে প্রথমে ১০ মান দেওয়া হয়েছে।
2. **print(n):** এই লাইনে চলক n এর মান প্রিন্ট করা হবে, যা ১০।
3. **n = 60:** এরপর n চলকের মান পরিবর্তন করা হচ্ছে ৬০।
4. **n = 70:** এরপর n এর মান আরও পরিবর্তিত হয়ে ৭০ হবে।
5. **n = 100:** অবশেষে, n এর মান ১০০ হয়ে যাবে।
6. **print(n):** দ্বিতীয় print(n) চলকের সর্বশেষ মান (১০০) প্রিন্ট করবে।

Assignment Operator:

= হলো **assignment operator**। এটি ডানপাশের মান (value) বা এক্সপ্রেশনকে বামপাশের চলক (variable) এর সাথে যুক্ত করে। একে চলকের মান নির্ধারণ বা আপডেট করার জন্য ব্যবহার করা হয়।

এখানে n চলকের মান বারবার পরিবর্তিত হয়েছে, এবং শেষের print(n) মান প্রিন্ট করার সময় n এর সর্বশেষ মান 100 হবে, কারণ assignment operator সবশেষে 100 মানটি n এ সংরক্ষণ করেছে।

❖ **Python comment:**

Python-এ কমেন্ট হলো এমন একটি অংশ যা কোডে কোনও কার্যকরী প্রভাব ফেলে না, বরং কোডের ব্যাখ্যা বা বিশদ তুলে ধরতে ব্যবহৃত হয়। কমেন্ট লেখার মাধ্যমে কোডের উদ্দেশ্য, কার্যকারিতা বা কোনো গুরুত্বপূর্ণ তথ্য বর্ণনা করা যায়, যা পরে কোড বুঝতে সহায়ক হয়।

```

1  # This is a single-line comment
2  print("Hello, NumPy!")
3
4
5  This is a multi-line comment.
6  Here we use single quotes.
7
8  print("Hello, Matplotlib!")
9  """
10 This is a multi-line comment.
11 Here we use double quotes.
12 """
13 print("Hello, Seaborn!")

```

Run new x

D:\PythonProject\PythonProject\.venv\Scripts\python.exe "C:\Users\Minhazul Kabir\Desktop\python01\new.py"

Hello, NumPy!
Hello, Matplotlib!
Hello, Seaborn!

Process finished with exit code 0

কমেন্ট লেখার উপায়:

1. **সিংগল লাইনের কমেন্ট:** # চিহ্ন দিয়ে এক লাইনের কমেন্ট লেখা হয়।
এই লাইনটি কমেন্ট
2. **মাল্টি-লাইন কমেন্ট:** একাধিক লাইনে কমেন্ট লেখার জন্য ট্রিপল কোটেশন (''' বা ''') ব্যবহার করা হয়।

```
'''
এটি মাল্টি-লাইন কমেন্ট
'''
```

কমেন্ট ব্যবহারের উদ্দেশ্য:

- **কোড ব্যাখ্যা:** কোডের কার্যকারিতা, পদ্ধতি বা লজিক বুঝানোর জন্য কমেন্ট ব্যবহার করা হয়।
- **ডিবাগিং:** কোডের কিছু অংশ অস্থায়ীভাবে নিষ্ক্রিয় (disable) করতে বা পরীক্ষা করার জন্য কমেন্ট করা যায়।
- **টিম ডেভেলপমেন্ট:** একাধিক ডেভেলপার একসাথে কাজ করলে, কোডে তাদের চিন্তা ও পরিকল্পনা ব্যাখ্যা করতে কমেন্ট সহায়ক হয়।

❖ **Performing type conversions in Python****Class Type:**

```

1 n=10
2 z=10.3
3 c="minhaz"
4 t= True
5 print(type(z))
6
7
8

```

Run new

```

D:\PythonProject\PythonProject\.venv\Scripts\python.exe "C:\Users\Minhazul Kabir\Desktop\python01\new.py"
<class 'float'>

Process finished with exit code 0

```

কোডের ব্যাখ্যা:

```

n = 10      # একটি পূর্ণসংখ্যা (integer) হিসেবে 10 মান দেয়া হচ্ছে
z = 10.3    # একটি দশমিক সংখ্যা (float) হিসেবে 10.3 মান দেয়া হচ্ছে

```

```
c = "minhaz" # একটি স্ট্রিং (string) হিসেবে "minhaz" মান দেয়া হচ্ছে
t = True     # একটি বুলিয়ান (boolean) মান দেয়া হচ্ছে, True
print(type(z)) # 'z' চলকের টাইপ প্রিন্ট করা হবে
```

এখানে `type(z)` ফাংশন কী করে?

`type()` ফাংশন একটি চলকের ডাটা টাইপ (data type) চিহ্নিত করে এবং সেটি ফিরিয়ে দেয়। এই কোডে, `z` একটি ফ্লোট (float) টাইপের চলক, কারণ এতে দশমিক মান (10.3) রয়েছে।

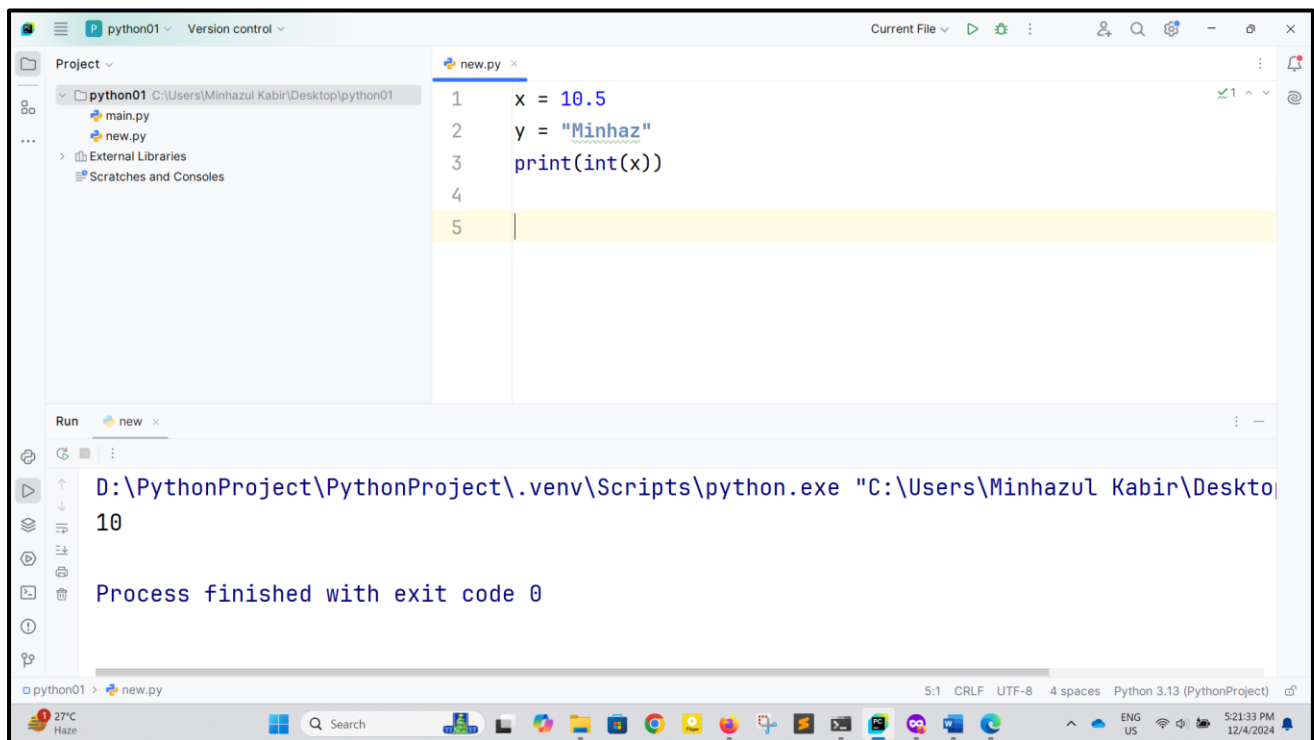
এটি `type(z)` দ্বারা `<class 'float'>` প্রিন্ট করবে, কারণ `z` ফ্লোট টাইপের।

ফলাফল:

```
<class 'float'>
```

এই ফলাফলটি বুঝায় যে `z` চলকটি একটি ফ্লোট টাইপের ডাটা ধারণ করছে।

Explicit Type Casting in Python:



Explicit Type Casting (প্রত্যক্ষ টাইপ কাস্টিং) হল একটি প্রক্রিয়া যেখানে আপনি নিজে একটি ডাটা টাইপকে অন্য ডাটা টাইপে রূপান্তরিত করেন। Python-এ এটি করতে সাধারণত `int()`, `float()`, `str()` ইত্যাদি ফাংশন ব্যবহার করা হয়। এটি তখন প্রয়োজন হয় যখন আপনি একটি ডাটা টাইপকে অন্য ডাটা টাইপে রূপান্তর করতে চান।

আপনার দেয়া কোডের ব্যাখ্যা:

```
x = 10.5      # float টাইপের মান
y = "Minhaz" # string টাইপের মান
print(int(x)) # explicit casting: float থেকে int এ কাস্ট করা হচ্ছে
```

কোডের বিশ্লেষণ:

1. `x = 10.5`:

- এখানে `x` একটি `float` টাইপের ভেরিয়েবল যার মান `10.5`।

2. `y = "Minhaz"`:

- এখানে `y` একটি `string` টাইপের ভেরিয়েবল যার মান `"Minhaz"`।

3. `print(int(x))`:

- `int(x)` এর মাধ্যমে `explicit type casting` করা হচ্ছে। এটি `x` (যা `10.5` একটি `float` টাইপ) কে `integer` টাইপে কাস্ট করছে।
- `int()` ফাংশনটি `float` থেকে `integer`-এ কাস্ট করে এবং দশমিক অংশ বাদ দেয়। সুতরাং, `10.5` থেকে কেবল `10` থাকবে, কারণ `int()` ফাংশন ফ্লোটের দশমিক অংশের মান (যেমন `.5`) সরিয়ে ফেলে।

ফলস্বরূপ:

- `int(10.5) → 10`

ফলাফল:

10

এটি `x` এর মান `10.5` কে `integer` টাইপে রূপান্তর করেছে এবং দশমিক অংশটি বাদ দিয়েছে, তাই `10` প্রিন্ট হয়েছে।

নোট:

- Explicit Type Casting** তখন ব্যবহার করা হয় যখন আপনি একটি নির্দিষ্ট টাইপে ডাটা কনভার্ট করতে চান। যেমন, `int()` ব্যবহার করে `float` কে `integer`-এ রূপান্তর করা, অথবা `str()` ব্যবহার করে `integer` বা `float` কে `string`-এ রূপান্তর করা।

এটি কীভাবে কাজ করে:

- `int()`: ফ্লোট বা স্ট্রিং টাইপকে পূর্ণসংখ্যায় (`integer`) রূপান্তর করে, এবং দশমিক অংশ বাদ দিয়ে কেবল পূর্ণসংখ্যা রাখে।
- এখানে, `x = 10.5` এর `int()` ফাংশন কল করলে `10.5` থেকে দশমিক অংশ বাদ দিয়ে `10` পাওয়া যায়।

যেকোনো ভুল কাস্টিং:

যদি আপনি একটি অগণনীয় স্ট্রিং (যেমন "Minhaz") `int()` ফাংশন দিয়ে কাস্ট করার চেষ্টা করেন, তাহলে `ValueError` হবে, কারণ "Minhaz" একটি সংখ্যায় রূপান্তরিত হতে পারে না।

```
y = "Minhaz"
```

```
print(int(y)) # এটি Error দিবে: ValueError: invalid literal for int() with base 10: 'Minhaz'
```

উপসংহার:

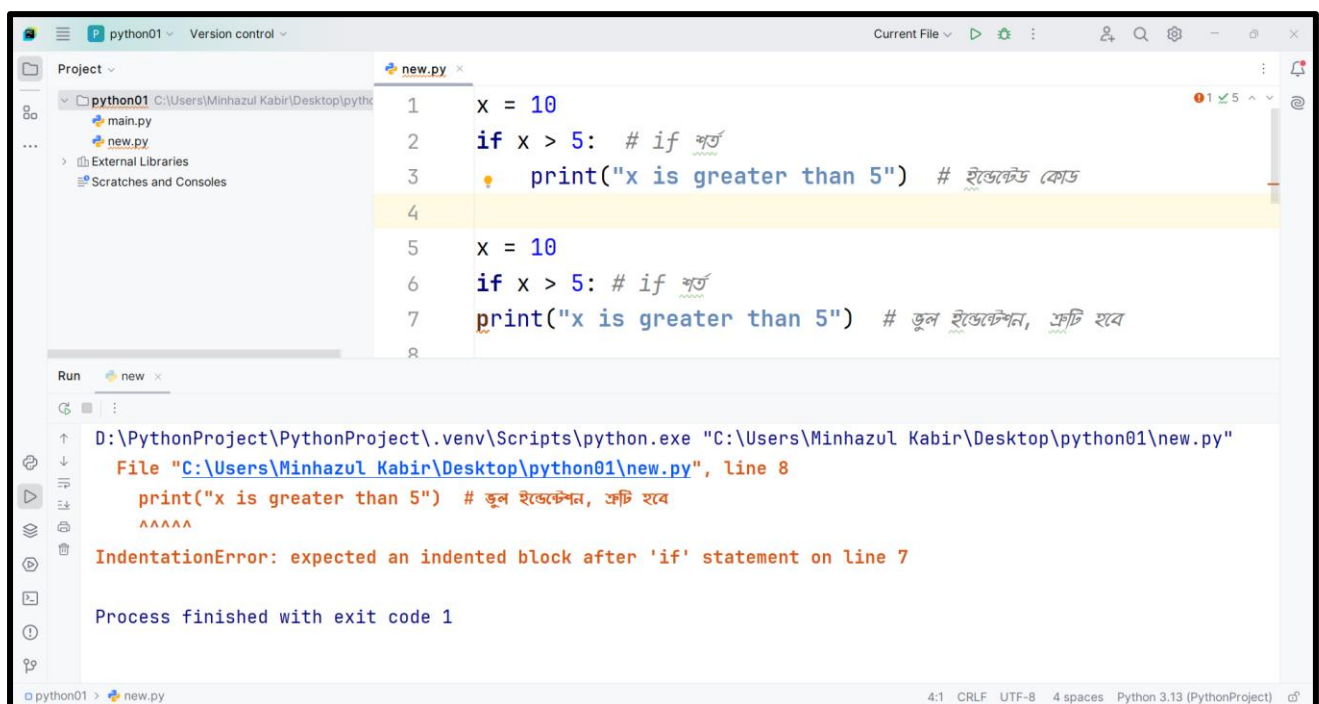
Explicit Type Casting হল Python-এ ডাটা টাইপ পরিবর্তন করার একটি গুরুত্বপূর্ণ প্রক্রিয়া। `int()`, `float()`, এবং `str()` ফাংশন ব্যবহার করে আপনি ডাটা টাইপ পরিবর্তন করতে পারেন। `int()` ফাংশন দিয়ে আপনি `float` থেকে `integer`-এ কাস্ট করতে পারেন, যেখানে দশমিক অংশ বাদ দেয়া হয়।

❖ Understanding basic Python syntax: statements, indentation, and comments.

1. Statements (স্টেটমেন্টস):

- **বোধগম্য উদাহরণ:** তুমি যদি তোমার বন্ধুকে বলো, "কাপড় চেঞ্জ করো", তাহলে এটি একটি স্টেটমেন্ট। এটি একটি নির্দেশনা যা বাস্তব জীবনে কিছু কাজ করার জন্য দেওয়া হয়। Python কোডে, একটি স্টেটমেন্ট হলো একটি নির্দেশনা যা কম্পিউটারকে কিছু কাজ করতে বলে, যেমন, কোনো মান নির্ধারণ করা বা কিছু প্রিন্ট করা।

2. Indentation (ইন্ডেন্টেশন):



```

1 x = 10
2 if x > 5: # if সত্য
3     print("x is greater than 5") # ইন্ডেন্টেড কোড
4
5 x = 10
6 if x > 5: # if সত্য
7 print("x is greater than 5") # ভুল ইন্ডেন্টেশন, সত্য হবে
8
Run new
D:\PythonProject\PythonProject\.venv\Scripts\python.exe "C:\Users\Minhazul Kabir\Desktop\python01\new.py"
File "C:\Users\Minhazul Kabir\Desktop\python01\new.py", line 8
    print("x is greater than 5") # ভুল ইন্ডেন্টেশন, সত্য হবে
    ^^^^^
IndentationError: expected an indented block after 'if' statement on line 7

Process finished with exit code 1
  
```

Indentation (ইন্ডেন্টেশন): ব্যবহার সুবিধা ও না ব্যবহারের অসুবিধা

ব্যবহারের সুবিধা:

1. কোডের পরিষ্কার গঠন: ইন্ডেন্টেশন কোডের ব্লক পরিষ্কারভাবে আলাদা করে, যা কোড বুঝতে সহায়ক।

উদাহরণ:

```
x = 10
if x > 5: # if শর্ত
    print("x is greater than 5") # ইন্ডেন্টেড কোড
```

2. সহজ ত্রুটি শনাক্তকরণ: ইন্ডেন্টেশন ভুল হলে Python ত্রুটি দেখায়, যা ডিবাগিং সহজ করে। উদাহরণ:

```
x = 10
if x > 5: # if শর্ত
print("x is greater than 5") # ভুল ইন্ডেন্টেশন, ত্রুটি হবে
```

3. Comments (কমেন্টস):

- বোধগম্য উদাহরণ: তুমি যখন একটি বই পড়ছো এবং বইয়ের মাঝে কোনো বিশেষ শব্দ বা ধারণা বোঝানোর জন্য তোমার নিজস্ব নোট লিখে রাখো, এটি যেমন তোমার জন্য সহায়ক, ঠিক তেমনি Python কোডে কমেন্টস হলো প্রোগ্রামারদের জন্য লেখা নোট যা কোডের কাজ ব্যাখ্যা করে, কিন্তু কম্পিউটার সেটি উপেক্ষা করে। এটি শুধু কোড লেখক বা দলের জন্য বোঝানোর উদ্দেশ্যে।

এই তিনটি ধারণা Python কোডিং এর মৌলিক স্তম্ভ, যেগুলো ছাড়া প্রোগ্রাম লেখা প্রায় অসম্ভব।

Practice:

The screenshot shows a Python IDE with a file named 'secondDay.py'. The code contains two examples of comments: one for a variable addition and one for string concatenation. The output window shows the results of these operations: 300 for the addition and '100200' for the string concatenation. The process finished with exit code 0.

```
1 # দুটি সংখ্যার যোগফল
2 num1 = 100 # প্রথম সংখ্যা
3 num2 = 200 # দ্বিতীয় সংখ্যা
4 print(num1 + num2) # আউটপুট হবে 300
5
6 # দুটি স্ট্রিংয়ের যোগফল (কনক্যাটেনেশন)
7 str1 = '100' # প্রথম স্ট্রিং
8 str2 = '200' # দ্বিতীয় স্ট্রিং
9 print(str1 + str2) # আউটপুট হবে '100200'
```

Run secondDay

```
"C:\Users\Minhazul Kabir\AppData\Local\Programs\Python\Python313\python.exe" "C:\Users\Minhazul Kabir\Desktop\python01\secondDay.py"
300
100200
Process finished with exit code 0
```

আপনার দেওয়া কোডে দুটি ভিন্ন প্রক্রিয়া আছে: একটিতে সংখ্যা (ইন্টিজার) যোগ করা হচ্ছে এবং অন্যটিতে স্ট্রিং (টেক্সট) যোগ করা হচ্ছে। এখানে পার্থক্য দুটি মূল বিষয় নিয়ে:

1. সংখ্যার যোগফল (Integer Addition):

```
num1 = 100 # প্রথম সংখ্যা  
num2 = 200 # দ্বিতীয় সংখ্যা  
print(num1 + num2) # আউটপুট হবে 300
```

- এখানে num1 এবং num2 দুটি **সংখ্যা (Integer)**।
- যখন দুইটি সংখ্যাকে যোগ করা হয়, তখন সেগুলোর যোগফল গোনা হয়।
- **ফলাফল:** $100 + 200 = 300$

2. স্ট্রিংয়ের যোগফল (String Concatenation):

```
str1 = '100' # প্রথম স্ট্রিং  
str2 = '200' # দ্বিতীয় স্ট্রিং  
print(str1 + str2) # আউটপুট হবে '100200'
```

- এখানে str1 এবং str2 দুটি **স্ট্রিং (টেক্সট)**।
- স্ট্রিং যোগ করার ক্ষেত্রে **কনক্যাটেনেশন** ঘটে, অর্থাৎ দুটি স্ট্রিং একত্রিত হয়ে একটি নতুন স্ট্রিং তৈরি হয়।
- **ফলাফল:** $'100' + '200' = '100200'$

পার্থক্য:

- **সংখ্যার যোগফল:** দুইটি সংখ্যার যোগফল হিসেবে তাদের গুণোত্তর পাওয়া যায়।
- **স্ট্রিংয়ের যোগফল:** দুইটি স্ট্রিং যোগ হলে তাদের সমষ্টির পরিবর্তে স্ট্রিংয়ের একত্রিত রূপ পাওয়া যায়।

সারাংশ:

- **সংখ্যার যোগফল** গণনা (Mathematical addition) হয়।
- **স্ট্রিংয়ের যোগফল** স্ট্রিং কনক্যাটেনেশন (Concatenation) হয়, অর্থাৎ টেক্সট যুক্ত করা হয়।