Day 2 Student Reference Sheet

Variables

```
x = 5
y = 3
print(x + y)

# This is a variable. It stores a value. Here we are adding two variables.
```

Data Types

```
name = "Ceren"
age = 17
height = 1.65
is_student = True
print(type(name), type(age), type(height), type(is_student))

# Python can store text, numbers, and even True/False values. These are called data types.
```

Lists

```
fruits = ["apple", "banana", "cherry"]
print(fruits[0])  # prints 'apple'

# Lists hold multiple items in a single variable.
```

Loops

```
for x in fruits:
    print("I like", fruit)

#A loop repeats a task for each item in a list.
```

If-Else

```
age = 18
if age >= 18:
    print("You can vote!")
else:
    print("Sorry, too young.")
```

```
You can vote!
```

Functions

```
def greet(name):
    return "Hello, " + name

print(greet("Murtaza"))

# Functions let us reuse code. This function greets a person.
```

Working with Data: NumPy Arrays and Basic Operations

Import NumPy

```
import numpy as np

#This imports NumPy. We use np as a standard shortcut.
```

Create an Array

```
my_array = np.array([1, 2, 3, 4, 5])
print(my_array)
```

```
my_array+5
```

Check Array Properties

```
print("Shape:", my_array.shape)
print("Data type:", my_array.dtype)
```

Perform Mathematical Operations

```
numbers = np.array([10, 20, 30,50])
print("Sum:", np.sum(numbers))
print("Mean:", np.mean(numbers))
print("Max:", np.max(numbers))
```

Math for Machine Learning: Linear Algebra Basics

Vectors – A one-dimensional array:

```
vector = np.array([2, 4, 6])
```

Matrices – A two-dimensional array:

```
matrix = np.array([[1, 2], [3, 4]])
```

Dot Product – Combines information from two vectors:

```
a = np.array([1, 2])
b = np.array([3, 4])

result = np.dot(a, b)
print("Dot product:", result)

# The dot product multiplies corresponding elements and adds them together. We'll see this again in Linear Regression!

# Dot Product of a and b = a1 x b1 + a2 x b2
```

Shape of Arrays:

```
print("Shape:",vector.shape)
print("Shape:", matrix.shape)
```

🧠 Calculus for ML: Gradients & Minimization

gradient example

```
import numpy as np

def f(x):
    return x**2

x = 3
h = 0.0001  # Small step size
gradient = (f(x + h) - f(x)) / h #formula (numerical derivative-turev)
print("Approximate gradient at x=3:", gradient)

#This code estimates the slope of the function f(x) = x² at x = 3."

# ✅ Key Idea:
# The gradient helps us figure out how to move in order to reduce the error.
```

📊 Statistics for ML: Mean, Variance, Standard Deviation

✅ The mean is the average.

✅ The variance measures how spread out the data is.

✅ The standard deviation tells us how much data deviates from the mean.

```
data = np.array([10, 12, 15, 18, 20])

print("Mean:", np.mean(data))  # Mean: The central point or average value of the data.
print("Variance:", np.var(data))  # Variance: Measures how much the data deviates from the mean.
print("Standard Deviation:", np.std(data))  # If standard deviation is high → data is widely spread. If low → data is tightl
```

Samples

```
grades = np.array([85, 90, 78, 92, 88])
print("Mean:", np.mean(grades))   # Output: 86.6
print("Standard Deviation:", np.std(grades))  # Output: ~5.0
```

```
ages = np.array([15, 16, 15, 17, 16])
print("Mean:", np.mean(ages))   # Output: 15.8
print("Standard Deviation:", np.std(ages))  # Output: ~0.74
```

```
steps = np.array([8000, 10000, 7500, 12000, 9500])
print("Mean:", np.mean(steps))   # Output: 9400.0
print("Standard Deviation:", np.std(steps))  # Output: ~1500.0
```

```
my_numbers = np.array([3, 8, 15, 22, 30])
print("Mean:", np.mean(my_numbers))   # Output: 15.6
print("Standard Deviation:", np.std(my_numbers))  # Output: ~9.2
```

Data in Action: Pandas Quick Intro + Linear Regression

🧱 Pandas Quick Intro

Pandas makes it easy to work with real-world datasets—things like CSV files, tables, and structured data.”

Step 1: Import Pandas

```
import pandas as pd
```

Step 2: Create a CSV Dataset in Colab (No Upload Needed!)

```
from io import StringIO

data = StringIO("""
Name,Age,Math,Science,English
Alice,16,85,90,88
Bob,17,78,84,75
Charlie,16,92,88,95
Ceren,16,86,98,56
Diana,15,70,75,80
Ethan,17,88,85,82
""")

df = pd.read_csv(data)
print(df.head())
```

Step 3: Explore the Data

```
print(df.columns)  # Show column names
print(df.info())   # Data types and non-null info
print(df.describe())  # Summary statistics
```

✅ Step 4: Access Columns & Rows

```
print(df['Age'])         # Access the 'Age' column
print(df.iloc[0])        # Access the first row
print(df['Math'].mean()) # Calculate the mean of the Math scores
```

✅ Mini Task for Students:

“Let’s try it together! Open a CSV file in Pandas, print the first 5 rows, and describe the data.”

```
import pandas as pd
from io import StringIO

# Create CSV data
data = StringIO("""
Name,Age,Math,Science,English
Alice,16,85,90,88
Bob,17,78,84,75
Ceren,16,86,98,56
Charlie,16,92,88,95
Diana,15,70,75,80
Ethan,17,88,85,82
""")

df = pd.read_csv(data)

# 1️⃣ Display the first 5 rows
print(df.head())

# 2️⃣ Show column names
print(df.columns)

# 3️⃣ Calculate the average of the 'Math' column
mean_math = df['Math'].mean()
print("Mean of Math column:", mean_math)
```

Colab Google Drive Eklentisi

```
from google.colab import drive
drive.mount('/content/drive')
```

Import a file from Google Drive

```
import pandas as pd
df = pd.read_csv('/content/drive/My Drive/TSU-AI-SUMMERCAMP-2025/ai-camp/students.csv')
print(df.head())
```

Listing Files and Viewing Column Names

```
!ls '/content/drive/My Drive/TSU-AI-SUMMERCAMP-2025/ai-camp'
print(df.columns)
```