Programming Error Cheat Sheet (COM4402)

Use this when you see an error.

1. Read the **LAST line** of the error.
2. Match it here by **error name**.
3. Apply the **fix checklist**.
4. Try to fix it yourself before asking for help.

1. Big Categories

Syntax error Python can't start (typo / missing symbol).

Runtime error Program starts but crashes while running.

Logic error No crash, but **wrong answer**.

Ask: **which one is this?**

2. Syntax Errors

2.1 `SyntaxError: invalid syntax`

Common causes:

- Missing colon:

```
if x > 0      # WRONG
if x > 0:     # RIGHT
```

- Unmatched brackets or quotes:

```
print("Hello)    # WRONG
print("Hello")   # RIGHT
```

- Typos in keywords or function names:

```
pritn("Hi")      # WRONG
print("Hi")      # RIGHT
```

Fix checklist:

- Check the line number in the error.
- Look on that line and the one above for:

- missing `:`
- missing `)`, `]`, `}`
- missing `"` or `'`
- spelling mistakes in `if`, `elif`, `else`, `print`, etc.

2.2 `IndentationError` / `TabError`

Example (WRONG):

```
if x > 0:
print("Positive")
```

Example (RIGHT):

```
if x > 0:
    print("Positive")
```

Meaning:

- Python expected an indented block, or your indentation is inconsistent.

Fix checklist:

- Use **4 spaces per indent level** (let PyCharm handle it).
- Make sure lines in the same block line up vertically.
- Avoid mixing tabs and spaces.

3. Name / Variable Errors

3.1 `NameError: name 'x' is not defined`

Meaning:

- You used a name Python doesn't know yet.

Common causes:

```
scor = 10
print(score)   # WRONG (score vs scor)
```

- Variable used before assignment.

Fix checklist:

- Check the spelling of variable/function names.
- Make sure the variable is assigned **before** it is used.

- Watch out for small differences: `age` vs `ages` vs `age_`.

4. Type Errors (Very Common)

4.1 `TypeError: unsupported operand type(s) for +: 'int' and 'str'`

Example (WRONG):

```
age = int(input("Enter age: "))
message = "Next year you will be " + age
```

Meaning:

- You tried to use `+` with a **string and a number**.

Fix options:

```
message = "Next year you will be " + str(age)
print("Next year you will be", age)
print(f"Next year you will be {age}")
```

Rule of thumb:

- `+` between strings → OK
- `+` between numbers → OK
- Don't mix without converting.

4.2 Doing Maths on Strings (Logic Error)

Example:

```
a = input("Enter first number: ")   # "2"
b = input("Enter second number: ")  # "3"
result = a + b                      # "23" not 5
```

Fix:

```
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
result = a + b
```

5. Value and Division Errors

5.1 `ValueError: invalid literal for int() with base 10`

Example:

```
age = int(input("Enter age: "))  # user types "twenty"
```

Meaning:

- Python tried to convert a string to `int`, but it wasn't a valid number.

Fix:

- When testing, type only digits (e.g. `19`, not `"nineteen"`).
- Later we'll add input validation; for now, check your input.

5.2 `ZeroDivisionError: division by zero`

Example:

```
total = 10
count = 0
average = total / count
```

Meaning:

- You tried to divide by zero.

Fix:

```
if count != 0:
    average = total / count
else:
    print("Error: cannot divide by zero")
```

6. Condition / Logic Mistakes

6.1 Using `=` instead of `==`

Example (WRONG):

```
if score = 100:
    print("Perfect")
```

This gives a `SyntaxError`.

Correct:

```
if score == 100:
    print("Perfect")
```

6.2 Grade Ordering Bug

Example (WRONG):

```
if score >= 40:
    grade = "Pass"
elif score >= 70:
    grade = "Distinction"
else:
    grade = "Fail"
```

Score `80` → `"Pass"` (first condition wins).

Correct (high to low):

```
if score >= 70:
    grade = "Distinction"
elif score >= 40:
    grade = "Pass"
else:
    grade = "Fail"
```

7. `input()` and Types

Golden rule: `input()` ALWAYS returns a **string**.

7.1 Forgetting to Convert Input

Example (WRONG):

```
age = input("Enter age: ")
age_next = age + 1
```

Gives `TypeError`.

Fix:

```
age = int(input("Enter age: "))
age_next = age + 1
```

7.2 Extra Spaces in Input

Example:

```
name = input("Enter name: ")  # user types "  Alice  "
```

Later:

```
if name == "Alice":
    ...
```

This might fail because of spaces.

Fix:

```
name = input("Enter name: ").strip()
```

8. `match` / `case` Basics (If You Use It)

Pattern:

```
match choice:
    case 1:
        print("Add")
    case 2:
        print("Subtract")
    case _:
        print("Invalid choice")
```

Common mistake:

- Forgetting `case _:` so **nothing happens** for other values.

9. Quick Debug Routine

Use this every time you get stuck:

1. **Read the last line** of the error message.
    - Example: `TypeError: unsupported operand type(s) for +: 'int' and 'str'`
2. Find that **error name** in this cheat sheet.
3. Apply the **fix checklist** in that section.
4. If it still fails, add `print()` to inspect values and types:

```
print(age, type(age))
```

5. Then ask for help, showing:
    - Your code
    - The full error message
    - Which section of this sheet you already checked