```
'''
Starting at 9:05 pm
Enjoy the song :)
'''


#Agenda
- Understand what Regular Expressions (regex) are and why they matter
- Learn core regex components (metacharacters, anchors, quantifiers, etc.)
- Practice matching patterns manually and via code
- Explore DevOps use cases: log parsing, masking, validation
- Work through hands-on Python examples
- Embedded quizzes for interactive learning


# What is Regex?
Regular Expression are patterns used
to match character combinations in strings.

# Why?
- Search, extract data from logs
- Validate email, phone, etc
- Automate text cleanup, masking, etc
- All progamming languages

#Devops use case?
- Kubernetes parsing or Docker logs
- Extract IPs, Timestamps
- Hide sensitive information in logs


\d{4}-\d{2}-\d{2} #regex

2025-09-08
\d -> digit
4,2,2 -> how many digits there can be


# Simple Matching & Metacharacters
'''
Common Metacharacters

Character       Meaning
.               Any character (except newline)  only one
\d              Digit (0-9)
\D              Not a digit
\w              Alphanumeric or underscore
\W              Not alphanumeric
\s              Whitespace
\S              Not whitespace
'''


"abc" -> .bc -> abc, xbc, bbc, 1bc, Abc, %bc all matched
'''
Any character is matched with .
```

```
a-z
A-Z
0-9
!@#$%^&*()_+
space, tab, punctuation
It wont match new line
'''
Doesnt match
bc -> no
aabc -> no


#Quiz 1
# Which of the following strings would match the pattern a.\d?
1: ab3 -> a.\d
2: a37 -> a.\d
3: a-  -> a.   wrong
4: a*7 -> a.\d
5: abc -> a.. wrong


# Quiz 2
# Which of the following strings would match the pattern ``\D\W\S`?
1: 3_a #not possible
2: @ #not possible
3: @#a -> \D\W\S
4: a-9 -> \D\W\S
5: 1& #not possible


# Anchors and Boundaries
'''
^abc – Matches if string starts with "abc".
This pattern checks whether the string begins with "abc".
'''
Matches:
"abc123"
"abcsdfghgeikghe"
"abc is the start of this"

Doesnt Match:
"xabcs"
"124abc"
" the abc"


'''
abc$ – Matches if string ends with "abc".
This pattern checks whether the string ends with "abc".
'''

Match:
"123abc"
"adjsgfawdoifnabc"
"ending with abc"
"aaaaabc"
```

```
Doesnt Match:
"abc124"
"xabcs"
"abc "

ing$
"Studying"
"working"
"just a beginning"
"just a ing"


'''
\bword\b – Matches the exact word "word" with word boundaries
(not as a part of another word).
This pattern matches "word" only when it appears as a
complete word,
not part of a longer word like "sword" or "wording".
'''

Match:
"this is a word"
"word,328940381@"
"the word is here"

Doesnt Match:
"password"
"sword"
"wording"



# Quiz 3
# Which string will match the pattern ^world?

" world starts here" - starting with space
"world-clock" -> ^world correct
"tworld" -> tstarting with t
"start with world"


# Quiz 4
# Which of the following strings will
# NOT match the regex \bsmile\b?
"the smile is key" -> match
"smile!" -> match
"a smile," -> match
"smileys are used a lot" -> not a match


Break: 10:00 - 10:10 pm


# Character Sets & Ranges
'''
```

```
Examples
[abc] → matches a, b, or c
[a-z] → matches any one lowercase letters
[0-9] → matches any one digits
[^a-z] → NOT lowercase letters
[A-Z] - Uppercase
'''

[A-Z][a-z][a-z]
one upercase
one lowercase
one lowercase


"App" -> match
"APo" -> has captial second letter hence no match
"Bob" -> match
"B1" -> no
"An" -> only one small char
"Ape" -> match


# Quiz 5
# What does [^a-z] match?
[a-z] -> all lowercase alphabets
[^a-z] -> anything but lowercase alphabets


# Quantifiers
'''
Symbol    Meaning
*           0 or more
+           1 or more
?           0 or 1
{3}       Exactly 3
{2,4}       Between 2 and 4
'''

# Groups
'''
() groups a sub-pattern.
This means you can treat part of the pattern as a single unit.
'''

"(ab)+" ->
ab
abababab
abaafjnsjkfnbab
abx
a -> doesnt match
acb -> doesnt match


"(abc)+" -> match abc as whole, it should occur 1 or more times
"abc+" -> ab whole, followed by 1 or more c
```

```
([\.-]?\w+)*

[\.-]? -> zero or one occurence of . or -
. is a metacharacter in regex hence using \.
- is not a metacharacter hence no need of \

\w+ ->
\w -> [a-z][A-Z][0-9]
+ -> one or more

[\.-]?\w+ -> optionally . or - followed by one or more
word characters

* -> complete group repearts 0 or more times

([\.-]?\w+)*

Match:
""
"hello"
"hello-world" -> hello + -world
"user.name" -> user + .name

Doesnt match:
"-" -> just hyphen no word after it
"."
"user..name" -> after first dot therse must be a word




#or symbol
(Mr | Mrs | Miss)


\b(Mr|Mrs|Ms)\.?\s[A-Z][a-z]+\b

\b -> word boundary
(Mr|Mrs|Ms) -> match wither of these
\.? -> optional .
\s -> one Whitespace
[A-Z] -> First better should be Capital letter
[a-z]+ -> any number of small letters
\b -> word boundary


"Mr. John"
"Mrs Smith"
"Ms. Alice"
"Hello Mr John"
"Ms Akanksha spoke to her students"

Doesnt match
```

"Mr hello John " -> not match
"mr john"
"Mr. JOHN"
"Ms."

Quiz 6
Which of the following strings would
#match the pattern \w+@\w+\.com?
\w+@\w+\.com

\w+ -> one or more word charcters
@ -> symbol @
\w+ -> one or more word charcters
\.com -> .com

word@word.com -> email like format

abc.xyz.com -> @ is missing
x_1@domain.co -> .co instead of .com
abc@xyz.com -> matching
@gmail.com -> starts with @ not a word

Quiz 7
Which regex matches at least 2 but max 4 digits?
\d{2,4}

Understanding Regex with an Online tool

https://regex101.com/

Pattern:
Pattern 1
Pattern: `\s(\d{3})\s`

Which part will this pattern match in the line?
```
127.0.0.1 - - [10/Oct/2023:13:56:55 +0000] "GET /index.html HTTP/1.1" 40
```

\s - whitespace
(\d{3}) -> 3 digits
\s whitespace
answer: 404

Pattern 2
What will be the regex for "Words of at least 5 letters"

Pattern 3
What do we use to match the whole word ?

```
What do we use to match the whole word ?


## Pattern 4
What will be regex for "First Letter Capital, Rest Lowercase"
```