```
import re

text = "Title: Mr. John"
pattern = r"(Mr|Mrs|Ms)\.?\s[A-Z][a-z]+"

match = re.search(pattern, text)

if match:
    print("Full Match:", match.group(0))   # Whole matched string
    print("Captured group:", match.group(1))   # Only the title (Mr,
    print("Captured group:", match.group(2))

else:
    print("No match found")
```

```
Full Match: Mr. John
Captured group: Mr
-------------------------------------------------------------------------
---
IndexError                                Traceback (most recent call
last)
/tmp/ipython-input-4045964536.py in <cell line: 0>()
      9     print("Full Match:", match.group(0))   # Whole matched
string
     10     print("Captured group:", match.group(1))   # Only the title
(Mr, Mrs, Ms)
---> 11     print("Captured group:", match.group(2))
     12
     13 else:

IndexError: no such group
```

Next steps: Explain error

```
import re

text = "Title: Mr. John"
pattern = r"(?:Mr|Mrs|Ms)\.?\s[A-Z][a-z]+"

match = re.search(pattern, text)
print("Full Match:", match.group(0))
print("Full Match:", match.group(1))
```

```
Full Match: Mr. John
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call
last)
```

```
#caturing groups
text="Date: 2025-08-21"
pattern=r"(\d{4})-(\d{2})-(\d{2})"
match=re.search(pattern,text)
print("Full",match.group(0))
print("Year",match.group(1))
print("Month",match.group(2))
print("Day",match.group(3))
```

```
Full 2025-08-21
Year 2025
Month 08
Day 21
```

```
#non capturing
text="Date: 2025-08-21"
pattern=r"(?:\d{4})-(?:\d{2})-(?:\d{2})"
match=re.search(pattern,text)
print("Full",match.group(0))
print("Year",match.group(1))
```

```
Full 2025-08-21
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call
last)
/tmp/ipython-input-2825364725.py in <cell line: 0>()
      4 match=re.search(pattern,text)
      5 print("Full",match.group(0))
----> 6 print("Year",match.group(1))

IndexError: no such group
```

Next steps: Explain error

```
(\d{4})-(\d{2})-(\d{2}) #capturing
(?:\d{4})-(?:\d{2})-(?:\d{2}) #non capturing
```

```
#manage file extensions

text="reports.pdf"
pattern=r".+\.(pdf|doc|txt)"
match=re.search(pattern,text)
print("Full:",match.group(0))
print("Extension:",match.group(1))
```

```
Full: reports.pdf
Extension: pdf
```

```
#manage file extensions

text="reports.pdf"
pattern=r"(.+)\.(pdf|doc|txt)"
match=re.search(pattern,text)
print("Full:",match.group(0))
print("File Name:",match.group(1))
print("Extension:",match.group(2))
```

```
Full: reports.pdf
File Name: reports
Extension: pdf
```

```
function -> purpose
'''
match - CHecks if the pattern matches only
the start of the string

search - search for the first match anywhere in the string

findall - return all the non overlapping matches as a list of string

finditer - return an iterator of match objects

sub - replace pattern matchers with a replacement string

split - split the string using the pattern as a delimiter
'''
```

```
data='''Mehul Sharma
615-555-7164
892 A-B Nagar
MehulSharma@bossmail.com

Vedavedya
634-345-2341
12th main street
Vedavedya@gmail.com

Eric Williams
560-555-5153
806 1st St., Faketown AK 86847
laurawilliams@bogusemail.com

Corey Jefferson
900-555-9340
826 Elm St., Epicburg NE 10671
coreyjefferson@bogusemail.com

Jennifer Martin-White
```

```
714-555-7405
212 Cedar St., Sunnydale CT 74983
jenniferwhite@bogusemail.com

Erick Davis
800-555-6771
519 Washington St."
'''

#extract the phone number
pattern=r"\d{3}-\d{3}-\d{4}"

#using match
print(re.match(pattern,data))


#using search
print(re.search(pattern,data))

#using findall
print(re.findall(pattern,data))

#using finditer()
numbers=re.finditer(pattern,data)
for i,num in enumerate(numbers):
  print(i)
  print(num)
  if(i==5):
    break
```

```
None
<re.Match object; span=(13, 25), match='615-555-7164'>
['615-555-7164', '634-345-2341', '560-555-5153', '900-555-9340', '714-555
0
<re.Match object; span=(13, 25), match='615-555-7164'>
1
<re.Match object; span=(76, 88), match='634-345-2341'>
2
<re.Match object; span=(141, 153), match='560-555-5153'>
3
<re.Match object; span=(231, 243), match='900-555-9340'>
4
<re.Match object; span=(328, 340), match='714-555-7405'>
5
<re.Match object; span=(417, 429), match='800-555-6771'>
```

```
import re
#validate 10-digit phone number
def is_valid_phone_number(s):
  pattern=r'\d{10}'
  return bool(re.fullmatch(pattern,s))

print(is_valid_phone_number("7346")) #false
print(is_valid_phone_number("7346670889")) #True
```

```
False
True
```

```
Break : 10:20 - 10:30 pm
```

```
#Extract Capitalized Words
import re
def get_Capital(text):
  return re.findall(r"\b[A-Z][a-z]*\b",text)
print(get_Capital("I study in Scaler Academcy is in Bangalore"))
```

```
['I', 'Scaler', 'Academcy', 'Bangalore']
```

```
#Extract Time Formats
import re
def extract_time(text):
  return re.findall(r"\d{2}:\d{2}(?::\d{2})?",text)

print(extract_time("Started at 09:05 ended at 11:15:05"))
```

```
['09:05', '11:15:05']
```

```
#Extract Words
def ending_words(text):
  return re.findall(r"\b\w+ing\b",text)

print(ending_words("helllo Vaibhav Running,testing, and logging is compl
```

```
['Running', 'testing', 'logging']
```

```
#Extract Error Codes
def extract_codes(text):
  return re.findall(r"\b(?:ERR|WARN|FAIL)\d+\b",text)

print(extract_codes("ERR123 occured after WARN456 and FAIL789"))
```

```
['ERR123', 'WARN456', 'FAIL789']
```

```
#Masking IP address
def mask_ip(text):
  return re.sub(r"\d{1,3}(?:\.\d{1,3}){3}","xxx",text)
print(mask_ip("Accessed 192.168.1.1 and 10.0.0.4"))
```

```
Accessed xxx and xxx
```

```
#Match email
#word@word.com
```

```
^\w+ -> start with a character
([\.-]?\w+)* -> optional
@\w+ -> domain name
(\.\w{2,3})+ -> domain suffix (.com, .co, .gov.in)

mahesh@xyz.com  match
@gmail.comp no
Ajay\ashid@domain no
akanksha.gaur-1@scaler.com match
```

```
def is_valid_email(s):
  pattern=r"^\w+([\.-]?\w+)*@\w+(\.\w{2,3})+$"
  return bool(re.search(pattern,s))

print(is_valid_email("akanksha.gaur-1@scaler.com"))
```

```
True
```

```
#ignore case
print(re.search("a+",'aaaAAAA'))
print(re.search("A+",'aaaAAAA'))
print(re.search("a+",'aaaAAAA',re.IGNORECASE))
print(re.search("A+",'aaaAAAA',re.IGNORECASE))
```

```
<re.Match object; span=(0, 3), match='aaa'>
<re.Match object; span=(3, 7), match='AAAA'>
<re.Match object; span=(0, 7), match='aaaAAAA'>
<re.Match object; span=(0, 7), match='aaaAAAA'>
```

```
#DOUBTS
```

```
#input in 2 D list
n,m=map(int,input().split()) #"4 5" -> n=4, m=5
matrix=[] #2Dlist

for row in range(n):
  One_D_list=list(map(int,input().split())) #"1 2 3 4 5" -> [1, 2, 3, 4,
  matrix.append(One_D_list)


print(matrix)
```

```
2 3
1 2 3
4 5 6
[[1, 2, 3], [4, 5, 6]]
```

```
#input in 2 D list using list compresion

n,m=map(int,input().split())
matrix=[list(map(int,input().split())) for _ in range(n)]
```

```
matrix=[list(map(int,input().split())) for _ in range(n)]
print(matrix)
```

```
3 4
3 4 5 6
3 4 5 6
2 1 3 2 4
[[3, 4, 5, 6], [3, 4, 5, 6], [2, 1, 3, 2, 4]]
```