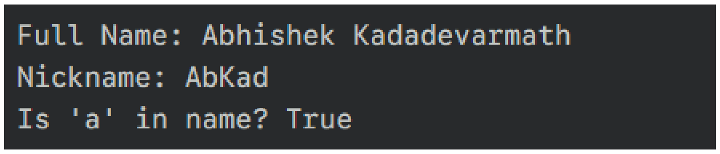
Exp-6: String Operations (Slicing, Concatenation, Searching, Formatting)

Name:Abhishek Kadadevarmath
Class: TY C
Roll No: C46

1. Read user's first and last name, create a nickname by slicing and concatenating parts of both names, search for a specific letter in the full name, and display using string formatting.

```
first = 'Abhishek'
last = 'Kadadevarmath'
full = first + ' ' + last
nickname = first[:2] + last[:3]  # e.g. YaKam
search_letter = 'a'
found = search_letter in full.lower()
print(f'Full Name: {full}')
print(f'Nickname: {nickname}')
print(f"Is '{search_letter}' in name? {found}")
```

Output:

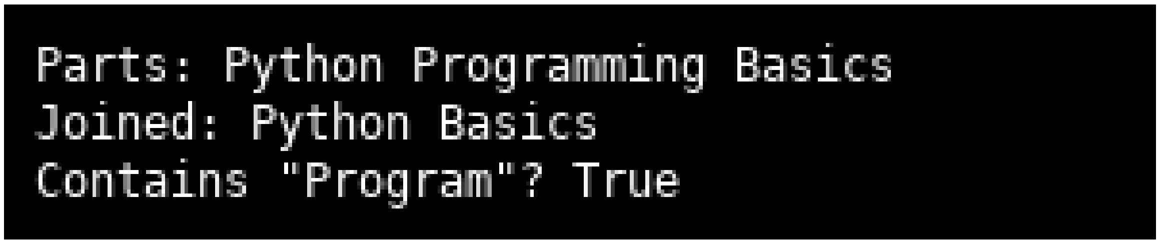


```
Full Name: Abhishek Kadadevarmath
Nickname: AbKad
Is 'a' in name? True
```

2. Process 'PythonProgrammingBasics', extract parts using slicing, join using concatenation, check if substring exists, display formatted result.

```
s = 'PythonProgrammingBasics'
part1 = s[:6]   # Python
part2 = s[6:17] # Programming
part3 = s[17:]  # Basics
joined = part1 + ' ' + part3
check = 'Program' in s
print('Parts:', part1, part2, part3)
print('Joined:', joined)
print('Contains "Program"?', check)
```

Output:

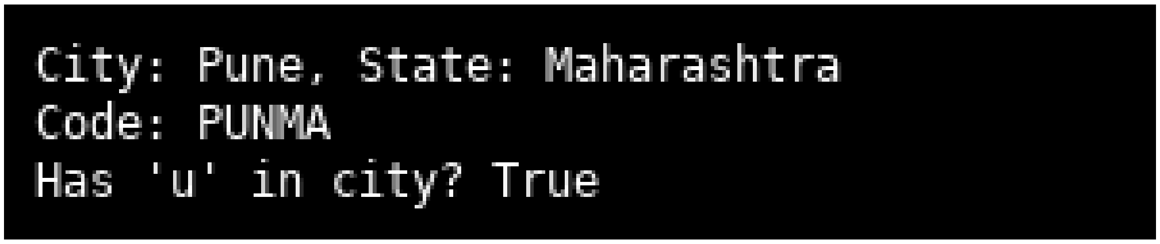


```
Parts: Python Programming Basics
Joined: Python Basics
Contains "Program"? True
```

3. Accept city and state names, generate a short code using slicing/concatenation, search for a character in city name, print formatted info.

```
city = 'Pune'
state = 'Maharashtra'
code = city[:3].upper() + state[:2].upper()
search_char = 'u'
has_char = search_char in city.lower()
print(f'City: {city}, State: {state}')
print(f'Code: {code}')
print(f"Has '{search_char}' in city? {has_char}")
```

Output:

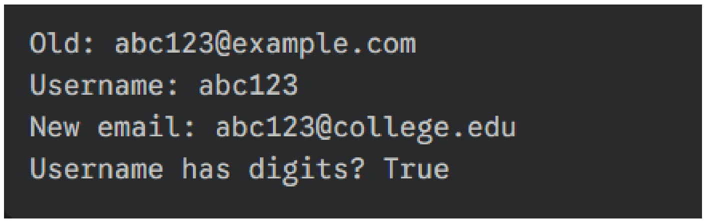


```
City: Pune, State: Maharashtra
Code: PUNMA
Has 'u' in city? True
```

4. Read an email, extract username before '@', concatenate with new domain, check for digits in username, display new email formatted.

```
email = 'abc123@example.com'
username = email.split('@')[0]
new_domain = 'college.edu'
new_email = username + '@' + new_domain
has_digit = any(ch.isdigit() for ch in username)
print('Old:', email)
print('Username:', username)
print('New email:', new_email)
print('Username has digits?', has_digit)
```

Output:

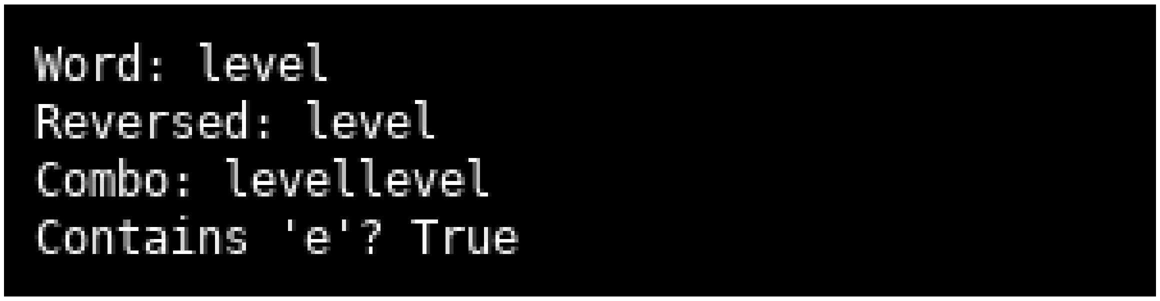


```
Old: abc123@example.com
Username: abc123
New email: abc123@college.edu
Username has digits? True
```

5. Take a word, reverse it using slicing, concatenate with original, search for matching character, and format output.

```
word = 'level'
rev = word[::-1]
combo = word + rev
search_c = 'e'
matches = search_c in combo
print('Word:', word)
print('Reversed:', rev)
print('Combo:', combo)
print(f"Contains '{search_c}'? {matches}")
```

Output:

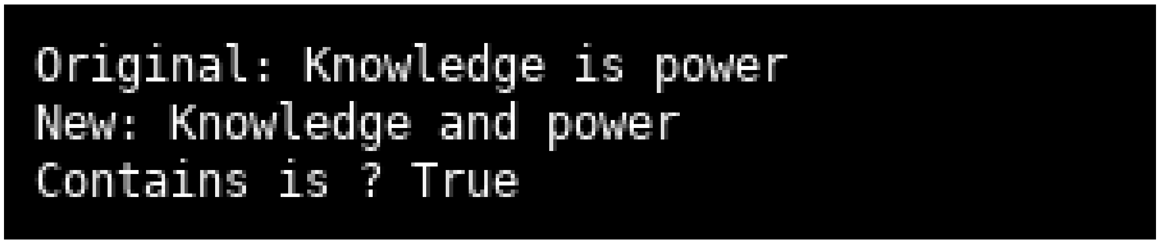


```
Word: level
Reversed: level
Combo: levellevel
Contains 'e'? True
```

6. Use 'Knowledge is power' to extract specific words with slicing, join them to form a new sentence, search for a given word, and print formatted sentence.

```
s = 'Knowledge is power'
words = s.split()
new_sentence = words[0] + ' and ' + words[2]
search_word = 'is'
print('Original:', s)
print('New:', new_sentence)
print('Contains', search_word, '?', search_word in s)
```

Output:

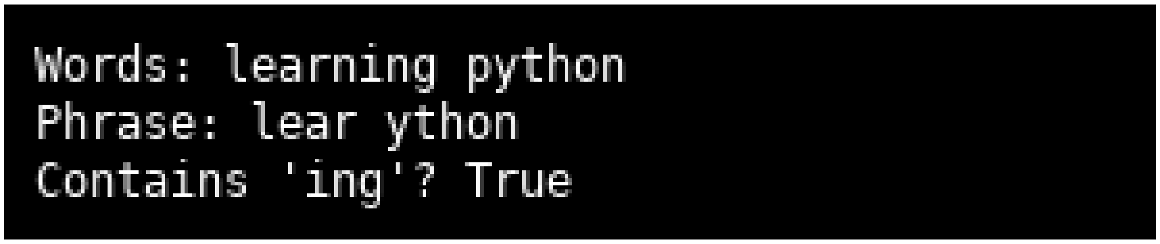


```
Original: Knowledge is power
New: Knowledge and power
Contains is ? True
```

7. Accept two words, extract portions using slicing, concatenate into a phrase, search for 'ing', display formatted result.

```
w1 = 'learning'
w2 = 'python'
part = w1[:4] + ' ' + w2[1:]
has_ing = 'ing' in w1 or 'ing' in w2
print('Words:', w1, w2)
print('Phrase:', part)
print("Contains 'ing'?", has_ing)
```

Output:

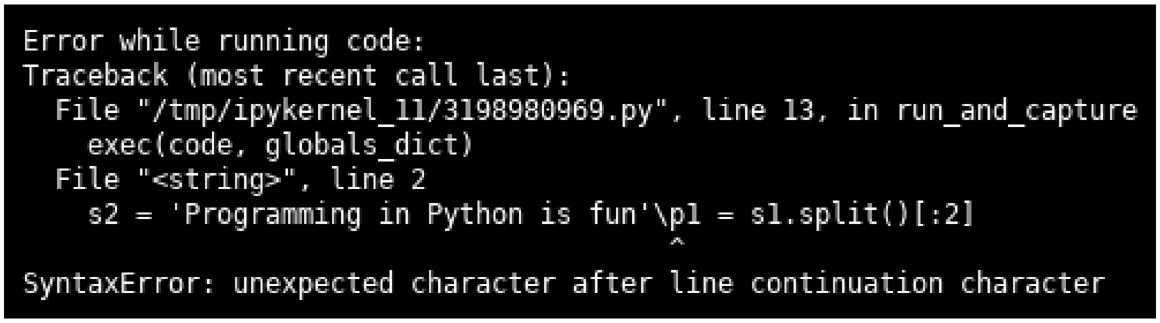


```
Words: learning python
Phrase: lear ython
Contains 'ing'? True
```

8. Accept two sentences, extract parts using slicing, combine them, search for a common substring, and print formatted comparison message.

```
s1 = 'I love programming'
s2 = 'Programming in Python is fun'\p1 = s1.split()[:2]
p2 = s2.split()[:3]
comb = ' '.join(p1) + ' & ' + ' '.join(p2)
common = any(word.lower() in s2.lower() for word in p1)
print('Combined:', comb)
print('Common word present?', common)
```

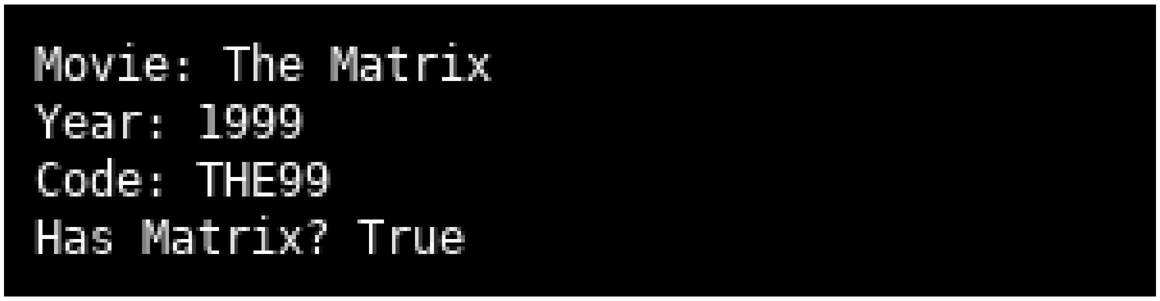
Output:


```
Error while running code:
Traceback (most recent call last):
  File "/tmp/ipykernel_11/3198980969.py", line 13, in run_and_capture
    exec(code, globals_dict)
  File "<string>", line 2
    s2 = 'Programming in Python is fun'\p1 = s1.split()[:2]
                                        ^
SyntaxError: unexpected character after line continuation character
```

9. Accept a movie name and release year, create a short movie code using slicing and concatenation, check for a specific word in the movie name, format final output.

```
movie = 'The Matrix'
year = '1999'
code = movie.replace(' ','')[:3].upper() + year[-2:]
has_word = 'Matrix' in movie
print('Movie:', movie)
print('Year:', year)
print('Code:', code)
print('Has Matrix?', has_word)
```

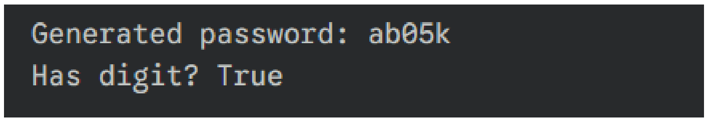
Output:


```
Movie: The Matrix
Year: 1999
Code: THE99
Has Matrix? True
```

10. Generate a simple password by slicing and concatenating parts of user's name and birth year, search for digits in it, and display the password formatted.

```
name = 'Abhishek'
year = '2005'
password = name[:2].lower() + year[-2:] + name[-1]
has_digit = any(ch.isdigit() for ch in password)
print('Generated password:', password)
print('Has digit?', has_digit)
```

Output:

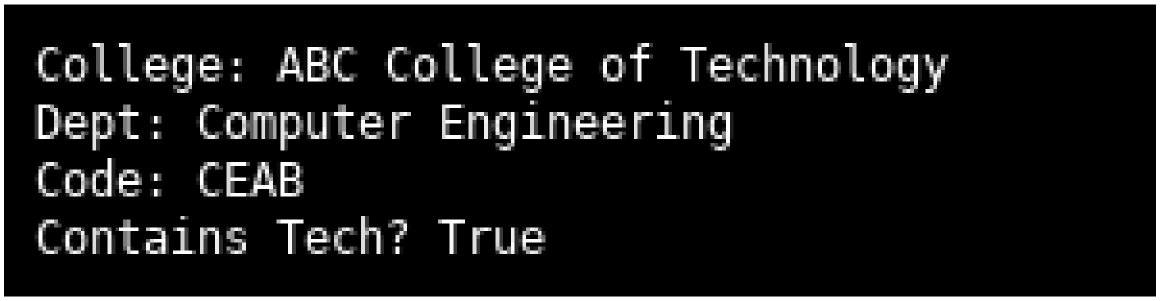


```
Generated password: ab05k
Has digit? True
```

11. Accept a college name and department, create department code using slicing/concatenation, search for 'Tech', and print formatted details.

```
college = 'ABC College of Technology'
dep = 'Computer Engineering'
code = ''.join([word[0] for word in dep.split()]) +
college.split()[0][:2]
has_tech = 'Tech' in college
print('College:', college)
print('Dept:', dep)
print('Code:', code)
print('Contains Tech?', has_tech)
```

Output:

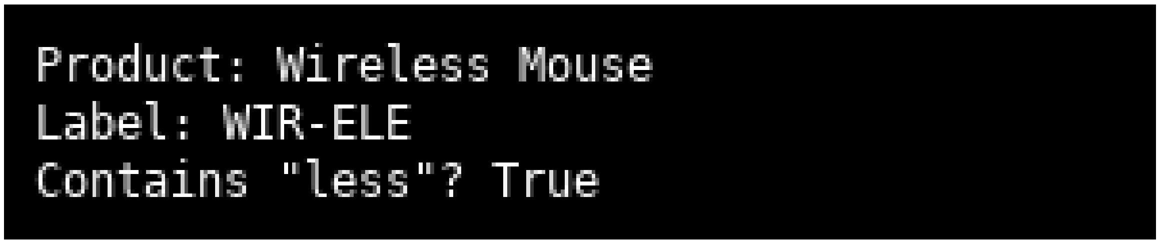


```
College: ABC College of Technology
Dept: Computer Engineering
Code: CEAB
Contains Tech? True
```

12. Read product name and category, form a label using sliced parts, search for a substring, and print the final label formatted.

```
product = 'Wireless Mouse'
category = 'Electronics'
label = product.split()[0][:3].upper() + '-' +
category[:3].upper()
search_sub = 'less' in product.lower()
print('Product:', product)
print('Label:', label)
print('Contains "less"?', search_sub)
```

Output:

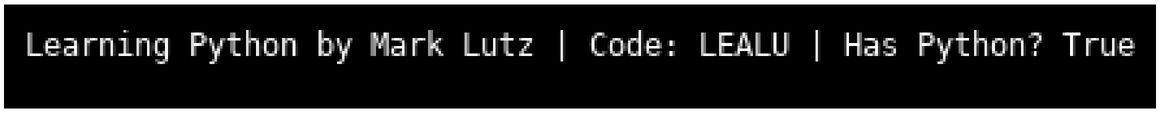


```
Product: Wireless Mouse
Label: WIR-ELE
Contains "less"? True
```

13. Input a book title and author, slice and join parts, search for a word in the title, and display info in single formatted string.

```
title = 'Learning Python'
author = 'Mark Lutz'
code = title.split()[0][:3].upper() +
author.split()[1][:2].upper()
has_word = 'Python' in title
print(f"{title} by {author} | Code: {code} | Has Python?
{has_word}")
```

Output:

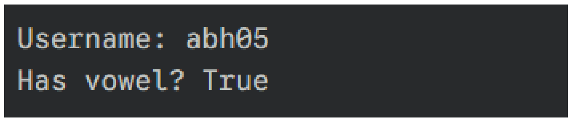


```
Learning Python by Mark Lutz | Code: LEALU | Has Python? True
```

14. Create social media username by slicing and concatenating parts of user's name and birth year, search for vowels in it, and display formatted username.

```
uname = 'Abhishek'
yob = '2005'
username = uname[:3].lower() + yob[-2:]
has_vowel = any(ch in 'aeiou' for ch in username)
print('Username:', username)
print('Has vowel?', has_vowel)
```

Output:

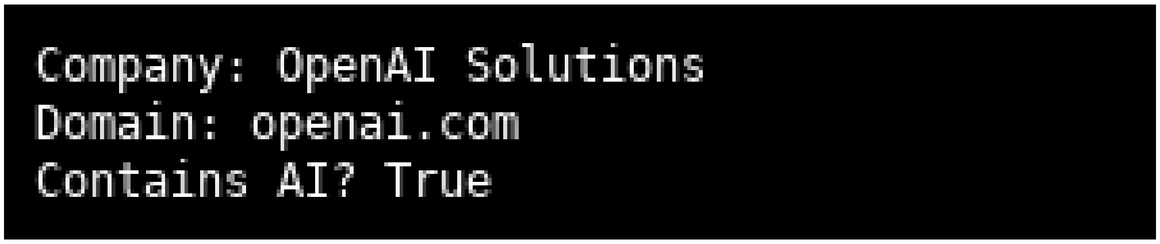


```
Username: abh05
Has vowel? True
```

15. Accept a company name, form a domain name by slicing and concatenation, check whether 'AI' exists in the name, and print the formatted domain details.

```
company = 'OpenAI Solutions'
domain = company.split()[0].lower() + '.com'
has_ai = 'AI' in company
print('Company:', company)
print('Domain:', domain)
print('Contains AI?', has_ai)
```

Output:



```
Company: OpenAI Solutions
Domain: openai.com
Contains AI? True
```