

APP WEEK 15

SYMBOLIC PROGRAMMING PARADIGM

Q1)

```
File Edit Selection View ...
app week 15

EXPLORER
APP WEEK 15
q1.py

q1.py > ...
1 import sympy
2
3 # Calculate the square root of 2 symbolically
4 sqrt_2 = sympy.sqrt(2)
5
6 # Convert the symbolic result to a decimal with 100 decimal places
7 sqrt_2_decimal = sqrt_2.evalf(100)
8
9 print(sqrt_2_decimal)
10
11

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python + - [ ] [ ] ... ^ x

PS C:\Users\Asus\Desktop\app week 15> C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q1.py"
1.414213562373095048801688724209698078569671875376948073176679737990732478462107038850387534327641573
PS C:\Users\Asus\Desktop\app week 15>
```

Q2)

```
File Edit Selection View ...
app week 15

EXPLORER
APP WEEK 15
q1.py
q2.py

q2.py > ...
1 import sympy
2
3 # Create rational numbers
4 half = sympy.Rational(1, 2)
5 third = sympy.Rational(1, 3)
6
7 # Perform the addition
8 result = half + third
9
10 print(result)
11

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python + - [ ] [ ] ... ^ x

PS C:\Users\Asus\Desktop\app week 15> C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q2.py"
5/6
PS C:\Users\Asus\Desktop\app week 15>
```

Q3)

```
1 import sympy
2
3 # Define the symbolic variables
4 x, y = sympy.symbols('x y')
5
6 # Define the expression
7 expression = (x + y)**6
8
9 # Expand the expression
10 expanded_expression = sympy.expand(expression)
11
12 print(expanded_expression)
13
```

PS C:\Users\Asus\Desktop\app week 15> C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q3.py"

$x^6 + 6x^5y + 15x^4y^2 + 20x^3y^3 + 15x^2y^4 + 6xy^5 + y^6$

PS C:\Users\Asus\Desktop\app week 15>

Q4)

```
1 import sympy
2
3 # Define the symbolic variable
4 x = sympy.symbols('x')
5
6 # Define the expression
7 expression = sympy.sin(x) / sympy.cos(x)
8
9 # Simplify the expression
10 simplified_expression = sympy.simplify(expression)
11
12 print(simplified_expression)
13
```

PS C:\Users\Asus\Desktop\app week 15> C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q4.py"

$\tan(x)$

PS C:\Users\Asus\Desktop\app week 15>

Q5)

```
2
3 # Define the symbolic variable
4 x = sympy.symbols('x')
5
6 # Define the expression
7 expression = (sympy.sin(x) - x) / x**3
8
9 # Calculate the limit as x approaches 0
10 limit_result = sympy.limit(expression, x, 0)
11
12 print(limit_result)
13
```

PS C:\Users\Asus\Desktop\app week 15> C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q6.py"

$-1/6$

Q6)

The screenshot shows a VS Code editor with a file explorer on the left containing files q1.py through q6(1).py under the 'APP WEEK 15' folder. The main editor displays q6(1).py, which uses SymPy to calculate derivatives. The code defines `derivative_sin` and `derivative_cos` using `sp.diff`, and prints the results for `log(x)`, `1/x`, `sin(x)`, and `cos(x)`. The terminal at the bottom shows the command `python.exe "c:/Users/Asus/Desktop/app week 15/q6(1).py"` and its output, which matches the printed results in the code.

```
10
11 # Derivative of sin(x) with respect to x
12 derivative_sin = sp.diff(sp.sin(x), x)
13
14 # Derivative of cos(x) with respect to x
15 derivative_cos = sp.diff(sp.cos(x), x)
16
17 print("Derivative of log(x):", derivative_log)
18 print("Derivative of 1/x:", derivative_1_over_x)
19 print("Derivative of sin(x):", derivative_sin)
20 print("Derivative of cos(x):", derivative_cos)
21
```

PS C:\Users\Asus\Desktop\app week 15> & C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q6(1).py"
Derivative of log(x): 1/x
Derivative of 1/x: -1/x**2
Derivative of sin(x): cos(x)
Derivative of cos(x): -sin(x)
PS C:\Users\Asus\Desktop\app week 15>

Q7)

The screenshot shows a VS Code editor with a file explorer on the left containing files q1.py through q7.py under the 'APP WEEK 15' folder. The main editor displays q7.py, which uses SymPy to solve a system of two linear equations. The code defines `eq1` and `eq2`, then uses `sp.solve` to find the solution for `x` and `y`. The terminal at the bottom shows the command `python.exe "c:/Users/Asus/Desktop/app week 15/q7.py"` and its output, which displays the solution: `x = -2` and `y = 4`.

```
4
5 # Define the system of equations
6 eq1 = sp.Eq(x + y, 2)
7 eq2 = sp.Eq(2*x + y, 0)
8
9 # Solve the system of equations
10 solution = sp.solve((eq1, eq2), (x, y))
11
12 print("Solution:")
13 print("x =", solution[x])
14 print("y =", solution[y])
15
```

PS C:\Users\Asus\Desktop\app week 15> & C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q7.py"
Solution:
x = -2
y = 4
PS C:\Users\Asus\Desktop\app week 15>

Q8)

The screenshot shows a VS Code editor with a file explorer on the left containing files q1.py through q8.py under the 'APP WEEK 15' folder. The main editor displays q8.py, which uses SymPy to calculate integrals. The code defines `integral_cos_x` and `integral_sin_x` using `sp.integrate`, and prints the results for `x^2`, `sin(x)`, and `cos(x)`. The terminal at the bottom shows the command `python.exe "c:/Users/Asus/Desktop/app week 15/q8.py"` and its output, which matches the printed results in the code.

```
11 # Integral of cos(x) with respect to x and y
12 integral_cos_x = sp.integrate(sp.cos(x), (x, 0, x), (y, 0, y))
13
14 print("Integral of x^2 with respect to x and y:")
15 print(integral_x_squared)
16
17 print("Integral of sin(x) with respect to x and y:")
18 print(integral_sin_x)
19
20 print("Integral of cos(x) with respect to x and y:")
21 print(integral_cos_x)
22
```

PS C:\Users\Asus\Desktop\app week 15> & C:/Python311/python.exe "c:/Users/Asus/Desktop/app week 15/q8.py"
Integral of x^2 with respect to x and y:
x**3*y/3
Integral of sin(x) with respect to x and y:
y*(1 - cos(x))
Integral of cos(x) with respect to x and y:
y*sin(x)
PS C:\Users\Asus\Desktop\app week 15>

Q9)

The screenshot shows the VS Code interface with the following components:

- Menu Bar:** File, Edit, Selection, View, ...
- Search Bar:** app week 15
- EXPLORER:**
 - APP WEEK 15
 - q1.py
 - q2.py
 - q3.py
 - q4.py
 - q6.py
 - q6(1).py
 - q7.py
 - q8.py
 - q9.py (selected)
- Editor:**

```

q9.py > ...
1  import sympy as sp
2
3  x = sp.symbols('x')
4  f = sp.Function('f')(x)
5
6  # Define the differential equation
7  diff_eq = sp.Eq(sp.Derivative(f, x) + 9*f, 1)
8
9  # Solve the differential equation
10 solution = sp.dsolve(diff_eq, f)
11
12 print("Solution to the differential equation:")
13 print(solution)

```
- TERMINAL:**

```

PS C:\Users\Asus\Desktop\app week 15> & C:/Python311/python.exe "c:/Users/Asus/Desktop/app
week 15/q9.py"
Solution to the differential equation:
Eq(f(x), C1*exp(-9*x) + 1/9)
PS C:\Users\Asus\Desktop\app week 15>

```

Q10)

The screenshot displays the Visual Studio Code interface. On the left, the Explorer sidebar shows a project named 'APP WEEK 15' containing files 'q1.py' through 'q10.py'. The main editor window shows the content of 'q10.py', which uses SymPy to solve a system of three linear equations. The code defines the equations, creates a system object, and solves it symbolically, printing the solution. The terminal at the bottom shows the command to run the script and the resulting output, which is a dictionary containing the values of x, y, and z.

```

1  q10.py > ...
2
3  8  eq1 = sp.Eq(3*x + 7*y - 12*z, c1)
4    eq2 = sp.Eq(4*x - 2*y - 5*z, c2)
5
6  10
7
8  11  # Create a system of equations
9    system = [eq1, eq2]
10
11  12
12  13
13  14  # Solve the system symbolically
14    solution = sp.solve(system, (x, y, z))
15
16  16
17  17  print("Solution:")
18    print(solution)
19

```

```

PS C:\Users\Asus\Desktop\app week 15> & C:/Python311/python.exe "c:/Users/Asus/Desktop/app
week 15/q10.py"
Solution:
{x: c1/17 + 7*c2/34 + 59*z/34, y: 2*c1/17 - 3*c2/34 + 33*z/34}

```