Q1. This is the first question: (**Each blank is worth 2 points, for a total of 10 points**)

```
cities = ["New York", "London", "Paris", "Tokyo", "Sydney", "Berlin", "Toronto"]
```

Using Python indexing and slicing techniques, fill in the blanks to extract the following values:

1. Extract "Tokyo" from the list using positive indexing:
   `city_tokyo = cities[__3__]`
2. Extract "Berlin" from the list using negative indexing:
   `city_berlin = cities[__-2__]`
3. Extract the sublist ["London", "Paris", "Tokyo"] using slicing:
   `city_sublist = cities[__1__ : __4__]`
4. Extract the last three cities using slicing and negative indexing:
   `last_three_cities = cities[__-3__ : ]`

Q2. This is the second question: (**Each blank is worth 1 points, for a total of 11 points**)

You are developing a data analysis tool, and a user has uploaded a CSV file containing weather data. After preprocessing, the data is stored in a multidimensional list `weather_data`, where each row represents a city's weekly temperatures. The data format is as follows:

```
weather_data = [
    ["City", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"],
    ["New York", 22, 24, 19, 21, 20, 18, 17],
    ["London", 15, 14, 17, 16, 14, 13, 12],
    ["Tokyo", 30, 32, 31, 33, 29, 28, 27],
    ["Sydney", 25, 24, 26, 23, 22, 21, 20]
]
```

Based on the following requirements, fill in the correct code to complete each task:

1. Extract Tokyo's Monday temperature:
   `tokyo_mon_temp = weather_data[__3__][__1__]`
2. Extract New York's Wednesday and Thursday temperatures and store them in a list:
   `nyc_midweek_temps = weather_data[__1__][__3__ : __5__]`
3. Extract London's entire weekly temperature data (excluding the city name):
   `london_week_temps = weather_data[__2__][__1__ : ]`
4. Using negative indexing, extract Sydney's Sunday temperature:
   `sydney_sun_temp = weather_data[__-1__][__-1__]`
5. Extract the Friday temperatures for all cities and generate a list `friday_temps`:
   `friday_temps = [weather_data[i][__5__] for i in range(__1__, len(weather_data))]`

Q3. This is the third question: (**The answer is worth 4 points**)

You are developing a weather monitoring system for a city. The system records the highest temperature for each day of the week, which is stored in separate variables. Your task is to check if any of the temperatures for the week exceeds 30°C. If so, print a heat warning. If none of the temperatures exceed 30°C, print a message indicating that the week was within safe temperature levels.

Here is the recorded temperature data for the week:

```
mon_temp = 28
tue_temp = 32
wed_temp = 29
thu_temp = 27
fri_temp = 31
sat_temp = 25
sun_temp = 26
```

What are the results of the following codes?

```
# List of temperatures for the week
weekly_temps = [mon_temp, tue_temp, wed_temp, thu_temp, fri_temp, sat_temp, sun_temp]

# Check if any temperature exceeds 30°C
if max(weekly_temps) > 30:
    print("Heat warning: Some days had temperatures above 30°C!")
else:
    print("The week had safe temperature levels.")
```

Solutions:

Heat warning: Some days had temperatures above 30°C!

Q4. This is the fourth question: (**The answer is worth 3 points**)

You are developing a user login system for a website. The system allows a user to attempt entering their password up to 3 times. If the password is correct, print a success message and exit the loop. If the password is incorrect, the user is asked to try again. If the user fails all three attempts, print a message denying access.

The correct password is stored as:

```
correct_password = "Python123"
```

The system will simulate user input by using a list of attempted passwords:

```
attempts = ["password", "Python12", "Python123"]
```

Write the code to simulate this login system using `while`, `if-else`, and indexing:

```
correct_password = "Python123"
attempts = ["password", "Python12", "Python123"]

# Initialize variables
attempt = 0
max_attempts = 3

# While loop for password attempts
while attempt < max_attempts:
    # Get the current password attempt using indexing
    user_input = attempts[attempt]

    if user_input == correct_password:
        print("Login successful!")
        break
    else:
        print(f"Attempt {attempt + 1}: Incorrect password. Try again.")

    # Increment the attempt counter
    attempt += 1

# If all attempts are used
if attempt == max_attempts and user_input != correct_password:
    print("Access denied. Too many incorrect attempts.")
```

Solutions:

```
Attempt 1: Incorrect password. Try again.
Attempt 2: Incorrect password. Try again.
Login successful!
```

Q5. This is the fifth question: (**Each blank is worth 2 points, for a total of 4 points**)

Fill in the blank to reverse the list `values = [1, 2, 3, 4, 5]` using slicing:
`reversed_values = values[_:_ : _-1_]`

Q6. This is the sixth question: (**Each blank is worth 2 points, for a total of 4 points**)

Given `fruits = ['apple', 'banana', 'cherry']`, modify the second item to 'mango':
`fruits[_1_] = '_mango_'`

Q7. This is the seventh question: (**Each blank is worth 1 points, for a total of 8 points**)

Given a list `` `[[1], [2, 3]]` ``, Please answer the following question:

1. If I want to convert this list into a NumPy array, what should I insert in the blank below, according to our lecture notes?

```
import numpy as np

numbers = np.__([[1], [2, 3]])
```

2. What is the result of the above codes?

Solutions:

```
1. numbers = np.array([[1], [2, 3]])

2. Raise an error. (Lists have different dimensions.)
```

Q8. This is the eighth question: (**Each blank is worth 2 points, for a total of 4 points**)

Given `letters = ['a', ['b', 'c', 'd'], 'e', 'f']`, how to get the letter 'd':
`result = letters[_1_][_2_]`

Q9. This is the ninth question: (**Each blank is worth 4 points, for a total of 4 points**)

Given `numbers = ['1', '2', '3', ['4', '5', '6', '7', '8']]`, the length of the list numbers is:
`length = _4_`

Q10. This is the ninth question: (**Each blank is worth 4 points, for a total of 4 points**)

Given `letters = ['a', 'b', 'c', ['d', 'e', ['f', 'g', 'h']]]`, the result of the `len[letters[-1][-1]]` is:
`result = _3_`

Q11. This is the eleventh question: (**Each blank is worth 4 points, for a total of 4 points**)

You are given a square matrix. Use Python indexing to extract the **main diagonal** (from top-left to bottom-right) as a list.

For example, given the matrix `matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]`, the output should be `[1, 5, 9]`.

```
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

# Extract the diagonal elements
diagonal = [matrix[i][i] for i in __]

print(diagonal)
```

Solutions:

```
matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

# Extract the diagonal elements
diagonal = [matrix[i][i] for i in range(len(matrix))]

print(diagonal)  # Output: [1, 5, 9]
```

Q12. This is the twelfth question: (**Each blank is worth 1 points, for a total of 3 points**)
You are tasked with developing a system to calculate an employee's weekly total salary. The base pay is $20 per hour, and overtime pay is $30 per hour for any hours worked over 40 in a week. Additionally, employees with a performance rating of 4 or above will receive a $200 bonus.

```
# 1. Input employee name, working hours, and performance rating
name = input("Enter employee's name: ")
hours = int(input("Enter hours worked this week: "))
rating = int(input("Enter employee's performance rating (1-5): "))

# 2. Calculate overtime hours
if hours > 40:
    overtime_hours = hours - 40
else:
    overtime_hours = ___

# 3. Calculate base and overtime pay
base_pay = 20 * min(hours, 40)
overtime_pay = ___ * overtime_hours
total_pay = base_pay + overtime_pay

# 4. Bonus for high performance
if rating >= 4:
    total_pay += ___

# 5. Output the result
print(f"Employee {name}'s total salary is: ${total_pay}")
```

Solutions:

```
# 1. Input employee name, working hours, and performance rating
name = input("Enter employee's name: ")
hours = int(input("Enter hours worked this week: "))
rating = int(input("Enter employee's performance rating (1-5): "))

# 2. Calculate overtime hours
if hours > 40:
    overtime_hours = hours - 40
else:
    overtime_hours = 0

# 3. Calculate base and overtime pay
base_pay = 20 * min(hours, 40)
overtime_pay = 30 * overtime_hours
total_pay = base_pay + overtime_pay

# 4. Bonus for high performance
if rating >= 4:
    total_pay += 200

# 5. Output the result
print(f"Employee {name}'s total salary is: ${total_pay}")
```

Q13. This is the thirteenth question: (**Each blank is worth 1 points, for a total of 3 points**)

A school library allows students to borrow up to 5 books, with each book having a maximum borrowing period of 30 days. If a book is overdue, a fine of $1 per day is charged. If a student has more than 2 overdue books, they cannot borrow more books until all overdue books are returned.

```
# 1. Input the number of books and the number of days each book has been borrowed
books = int(input("Enter the number of books borrowed: "))
days = [int(input(f"Enter the number of days for book {i+1}: ")) for i in range(books)]

# 2. Calculate the number of overdue books and the total fine
late_books = 0
fine = 0
for day in days:
    if day > ___:
        late_books += 1
        fine += (day - 30) * ___

# 3. Determine if the student can borrow more books
if late_books > ___:
    can_borrow_more = False
else:
    can_borrow_more = True

# 4. Output the fine and borrowing status
print(f"Total fine: ${fine}")
if can_borrow_more:
    print("The student can borrow more books.")
else:
    print("The student cannot borrow more books due to overdue books.")
```

Solutions:

```
# 1. Input the number of books and the number of days each book has been borrowed
books = int(input("Enter the number of books borrowed: "))
days = [int(input(f"Enter the number of days for book {i+1}: ")) for i in range(books)]

# 2. Calculate the number of overdue books and the total fine
```

```
late_books = 0
fine = 0
for day in days:
    if day > 30:
        late_books += 1
        fine += (day - 30) * 1

# 3. Determine if the student can borrow more books
if late_books > 2:
    can_borrow_more = False
else:
    can_borrow_more = True

# 4. Output the fine and borrowing status
print(f"Total fine: ${fine}")
if can_borrow_more:
    print("The student can borrow more books.")
else:
    print("The student cannot borrow more books due to overdue books.")
```

Q14. This is the fourteenth question: (**Each blank is worth 1 points, for a total of 3 points**)

You are developing a hotel booking system. The hotel offers three types of rooms:

- Single room: $50 per night
- Double room: $80 per night
- Luxury suite: $150 per night

Guests can add extra services:

- Breakfast: $10 per day
- Gym access: $15 per day

If a guest stays for 5 or more days, they receive a 10% discount on the room rate.

```
# 1. Select room type and number of nights
room_type = input("Enter room type (Single/Double/Luxury): ")
days = int(input("Enter number of nights: "))

# 2. Set room price based on the type
if room_type == "Single":
    room_price = ___
elif room_type == "Double":
    room_price = 80
elif room_type == "Luxury":
    room_price = 150
```

```
else:
    room_price = 0

# 3. Check for extra services
breakfast = input("Would you like breakfast? (Yes/No): ") == "Yes"
gym = input("Would you like gym access? (Yes/No): ") == "Yes"

# 4. Calculate total cost
total_price = room_price * days
if breakfast:
    total_price += 10 * days
if gym:
    total_price += ___

# 5. Apply discount if applicable
if days >= ___:
    total_price *= 0.9

# 6. Output the total cost
print(f"Total cost: ${total_price}")
```

Solutions:

```
# 1. Select room type and number of nights
room_type = input("Enter room type (Single/Double/Luxury): ")
days = int(input("Enter number of nights: "))

# 2. Set room price based on the type
if room_type == "Single":
    room_price = 50
elif room_type == "Double":
    room_price = 80
elif room_type == "Luxury":
    room_price = 150
else:
    room_price = 0

# 3. Check for extra services
breakfast = input("Would you like breakfast? (Yes/No): ") == "Yes"
gym = input("Would you like gym access? (Yes/No): ") == "Yes"

# 4. Calculate total cost
total_price = room_price * days
if breakfast:
    total_price += 10 * days
if gym:
    total_price += 15 * days
```

```
# 5. Apply discount if applicable
if days >= 5:
    total_price *= 0.9

# 6. Output the total cost
print(f"Total cost: ${total_price}")
```

Q15. This is the fifteenth question: (**Each blank is worth 1 points, for a total of 3 points**)
Manage student grades where average scores below 60 require retakes, scores between 60 and 80 are satisfactory, and above 80 are excellent. Courses with scores below 50 require retakes.

```
# 1. Input student name and grades for multiple subjects
student_name = input("Enter the student's name: ")
grades = [int(input(f"Enter grade for subject {i+1}: ")) for i in
range(5)]

# 2. Calculate average grade
average = sum(grades) / ___

# 3. Determine the performance level
if average < 60:
    result = "Needs Improvement"
elif ___ <= average < 80:
    result = "Satisfactory"
else:
    result = "Excellent"

# 4. Identify subjects that require a retake
retake_subjects = [i+1 for i, grade in enumerate(grades) if grade < ___]

# 5. Output the result
print(f"{student_name}'s average score is: {average}")
print(f"Performance level: {result}")
if retake_subjects:
    print(f"Subjects that need retake: {retake_subjects}")
```

Solutions:

```
# 1. Input student name and grades for multiple subjects
student_name = input("Enter the student's name: ")
grades = [int(input(f"Enter grade for subject {i+1}: ")) for i in
range(5)]

# 2. Calculate average grade
average = sum(grades) / 5

# 3. Determine the performance level
if average < 60:
    result = "Needs Improvement"
elif 60 <= average < 80:
    result = "Satisfactory"
else:
    result = "Excellent"

# 4. Identify subjects that require a retake
retake_subjects = [i+1 for i, grade in enumerate(grades) if grade < 50]

# 5. Output the result
print(f"{student_name}'s average score is: {average}")
print(f"Performance level: {result}")
if retake_subjects:
    print(f"Subjects that need retake: {retake_subjects}")
```

Q16. This is the sixteenth question: (**Each blank is worth 1 points, for a total of 4 points**)

An online store needs a shopping cart system. The system allows customers to add items to their cart and calculates the total price. If the total price exceeds $200, the customer receives a 15% discount. Shipping costs $10 for orders under $50, but it's free for orders above $50. Additionally, if the customer orders more than 3 items, they get a special free gift.

```
# 1. Input the number of items and their prices
cart = []
for i in range(int(input("Enter the number of items: "))):
    price = float(input(f"Enter the price for item {i+1}: "))
    cart.append(price)

# 2. Calculate total price
total_price = sum(cart)

# 3. Apply discount if total price exceeds $200
if total_price > ___:
    total_price *= ___

# 4. Determine shipping cost
if total_price < ___:
    shipping = 10
else:
    shipping = 0

# 5. Check for free gift
if len(cart) > ___:
    gift = "Free Gift"
else:
    gift = "No Gift"

# 6. Output the final price and gift status
print(f"Total price: ${total_price + shipping}")
print(f"Shipping cost: ${shipping}")
print(f"Gift: {gift}")
```

Solutions:

```
# 1. Input the number of items and their prices
cart = []
for i in range(int(input("Enter the number of items: "))):
    price = float(input(f"Enter the price for item {i+1}: "))
    cart.append(price)

# 2. Calculate total price
total_price = sum(cart)

# 3. Apply discount if total price exceeds $200
if total_price > 200:
    total_price *= 0.85

# 4. Determine shipping cost
if total_price < 50:
    shipping = 10
else:
    shipping = 0

# 5. Check for free gift
if len(cart) > 3:
    gift = "Free Gift"
else:
    gift = "No Gift"

# 6. Output the final price and gift status
print(f"Total price: ${total_price + shipping}")
print(f"Shipping cost: ${shipping}")
print(f"Gift: {gift}")
```

Q17. This is the seventeenth question: (**Each blank is worth 2 points, for a total of 10 points**)

A restaurant wants a system to calculate customer bills. Customers can order from the menu, which includes items with different prices. There is a 10% service charge on the total bill, and customers can optionally leave a tip. If the total bill exceeds $100, the service charge is waived.

```
# 1. Define the menu and prices
menu = {"burger": 12.0, "pizza": 15.0, "salad": 8.0, "soda": 3.0}

# 2. Take orders
orders = []
for i in range(int(input("Enter number of items ordered: "))):
    item = input(f"Enter the name of item {i+1}: ")
    if item in menu:
        orders.append(___)
    else:
        print(f"{item} is not on the menu.")

# 3. Calculate total bill
total_bill = sum(orders)

# 4. Apply service charge if the bill is under $100
if total_bill <= ___:
    service_charge = total_bill * 0.1
else:
    service_charge = 0

# 5. Ask for a tip
tip = float(input("Enter the tip amount (or 0 for no tip): "))

# 6. Calculate and display final bill
final_bill = _________ + ____________+ ___
print(f"Total bill: ${final_bill} (Service charge: ${service_charge})")
```

Solutions:

```
# 1. Define the menu and prices
menu = {"burger": 12.0, "pizza": 15.0, "salad": 8.0, "soda": 3.0}

# 2. Take orders
orders = []
for i in range(int(input("Enter number of items ordered: "))):
    item = input(f"Enter the name of item {i+1}: ")
    if item in menu:
        orders.append(menu[item])
    else:
        print(f"{item} is not on the menu.")

# 3. Calculate total bill
total_bill = sum(orders)
```

```
# 4. Apply service charge if the bill is under $100
if total_bill <= 100:
    service_charge = total_bill * 0.1
else:
    service_charge = 0

# 5. Ask for a tip
tip = float(input("Enter the tip amount (or 0 for no tip): "))

# 6. Calculate and display final bill
final_bill = total_bill + service_charge + tip
print(f"Total bill: ${final_bill} (Service charge: ${service_charge})")
```

Q18. This is the eighteenth question: (**Each blank is worth 1 points, for a total of 3 points**)

A university course registration system needs to allow students to register for courses. The maximum number of courses a student can register for is 6. If a student registers for more than 4 courses, they receive a discount on the total tuition. The cost per course is $300.

```
# 1. Input the number of courses the student wants to register for
courses = int(input("Enter the number of courses: "))

# 2. Ensure the student is not registering for more than 6 courses
if courses > ___:
    print("You cannot register for more than 6 courses.")
else:
    # 3. Calculate the total tuition
    tuition = courses * ___

    # 4. Apply a discount if more than 4 courses are registered
    if courses > ___:
        tuition *= 0.9

    # 5. Output the final tuition cost
    print(f"Total tuition cost: ${tuition}")
```

Solutions:

```
# 1. Input the number of courses the student wants to register for
courses = int(input("Enter the number of courses: "))

# 2. Ensure the student is not registering for more than 6 courses
if courses > 6:
    print("You cannot register for more than 6 courses.")
else:
    # 3. Calculate the total tuition
    tuition = courses * 300

    # 4. Apply a discount if more than 4 courses are registered
    if courses > 4:
        tuition *= 0.9

    # 5. Output the final tuition cost
    print(f"Total tuition cost: ${tuition}")
```

Q19. This is the nineteenth question: (**Each blank is worth 1 points, for a total of 5 points**)

A fitness center needs a membership system. The center offers three types of memberships:

- Basic: $30 per month
- Premium: $50 per month
- VIP: $80 per month

If a member pays for more than 6 months upfront, they get a 10% discount on the total price. The center also offers personal training sessions at $20 per session.

```
# 1. Input the type of membership and number of months
membership_type = input("Enter membership type (Basic/Premium/VIP): ")
months = int(input("Enter the number of months: "))

# 2. Set membership price based on type
if membership_type == "Basic":
    price = ___
elif membership_type == "Premium":
    price = ___
elif membership_type == "VIP":
    price = ___
else:
    price = 0

# 3. Apply discount if more than 6 months
if months > ___:
    total_price = price * months * 0.9
else:
    total_price = price * months

# 4. Ask about personal training sessions
sessions = int(input("Enter number of personal training sessions: "))
total_price += ___ * sessions

# 5. Output the total membership cost
```

```
print(f"Total membership cost: ${total_price}")
```

Solutions:

```
# 1. Input the type of membership and number of months
membership_type = input("Enter membership type (Basic/Premium/VIP): ")
months = int(input("Enter the number of months: "))

# 2. Set membership price based on type
if membership_type == "Basic":
    price = 30
elif membership_type == "Premium":
    price = 50
elif membership_type == "VIP":
    price = 80
else:
    price = 0

# 3. Apply discount if more than 6 months
if months > 6:
    total_price = price * months * 0.9
else:
    total_price = price * months

# 4. Ask about personal training sessions
sessions = int(input("Enter number of personal training sessions: "))
total_price += 20 * sessions

# 5. Output the total membership cost
print(f"Total membership cost: ${total_price}")
```

Q20. This is the twentieth question: (**Each blank is worth 1 points, for a total of 6 points**)

A bank offers loans with varying interest rates based on the loan amount. For loans under $10,000, the interest rate is 5%. For loans between $10,000 and $20,000, the rate is 3%. For loans over $20,000, the rate is 2%. The customer can choose the loan term (in years), and the total amount payable is calculated based on the loan amount, interest rate, and term.

```
# 1. Input loan amount and loan term
loan_amount = float(input("Enter the loan amount: "))
years = int(input("Enter the loan term (in years): "))

# 2. Set interest rate based on loan amount
if loan_amount < ___:
    interest_rate = ___
elif loan_amount <= ___:
    interest_rate = ___
else:
    interest_rate = ___

# 3. Calculate total interest
total_interest = loan_amount * (interest_rate / 100) * years

# 4. Calculate total payable amount
total_payable = loan_amount + ___

# 5. Output the total payable amount
print(f"Total amount payable: ${total_payable}")
```

Solutions:

```
# 1. Input loan amount and loan term
loan_amount = float(input("Enter the loan amount: "))
years = int(input("Enter the loan term (in years): "))

# 2. Set interest rate based on loan amount
if loan_amount < 10000:
    interest_rate = 5
elif loan_amount <= 20000:
    interest_rate = 3
else:
    interest_rate = 2

# 3. Calculate total interest
total_interest = loan_amount * (interest_rate / 100) * years

# 4. Calculate total payable amount
total_payable = loan_amount + total_interest

# 5. Output the total payable amount
print(f"Total amount payable: ${total_payable}")
```