Functions, Variables

1. **Calculate the factorial of a number.**
   - Hint: Use recursion.
2. **Find the largest of three numbers.**
   - Hint: Use nested if statements.
3. **Convert temperatures from Celsius to Fahrenheit.**
   - Hint: Use the formula F=95C+32F
4. **Create a function to check if a number is prime.**
   - Hint: Loop from 2 to the square root of the number.
5. **Write a function to reverse a string.**
   - Hint: Use string slicing.
6. **Create a function to compute the nth Fibonacci number.**
   - Hint: Use recursion or iteration.
7. **Write a function to calculate the area of a circle.**
   - Hint: Use the formula A=πr2A = \pi r^2A=πr2.
8. **Implement a function to sort a list of numbers.**
   - Hint: Use a sorting algorithm like bubble sort or Python's built-in `sorted()`.
9. **Create a function to find the GCD of two numbers.**
   - Hint: Use the Euclidean algorithm.
10. **Write a function to check if a string is a palindrome.**
    - Hint: Compare the string with its reverse.
11. **Create a function that takes a list and returns a new list with unique elements.**
    - Hint: Use a set to filter duplicates.
12. **Write a function to calculate the sum of squares of the first n natural numbers.**
    - Hint: Use a loop or formula n(n+1)(2n+1)6\frac{n(n+1)(2n+1)}{6}6n(n+1)(2n+1) .
13. **Create a function to merge two sorted lists.**
    - Hint: Use the merge step of merge sort.
14. **Write a function to find the second largest number in a list.**
    - Hint: Traverse the list while keeping track of the two largest numbers.
15. **Implement a function to remove vowels from a string.**
    - Hint: Use a list comprehension to filter out vowels.
16. **Create a function that returns the transpose of a matrix.**
    - Hint: Use nested loops or list comprehensions.
17. **Write a function to flatten a nested list.**
    - Hint: Use recursion or a stack.
18. **Create a function to find the longest common prefix of a list of strings.**
    - Hint: Compare characters one by one.
19. **Write a function to check if two strings are anagrams.**
    - Hint: Sort both strings and compare.
20. **Implement a function to calculate the dot product of two vectors.**
    - Hint: Use the sum of products of corresponding elements.

Conditionals

1. **Check if a given year is a leap year.**
    - Hint: Use the rules for leap years (divisible by 4, not by 100 unless by 400).
2. **Determine the grade of a student based on their score.**
    - Hint: Use if-elif-else statements.
3. **Find the sign of a given number (positive, negative, or zero).**
    - Hint: Use nested if statements.
4. **Check if a number is even or odd.**
    - Hint: Use the modulus operator.
5. **Determine if a character is a vowel or consonant.**
    - Hint: Use a list or set of vowels.
6. **Check if a string is a valid email address.**
    - Hint: Use basic string operations or regular expressions.
7. **Determine if a point lies inside a rectangle given its coordinates.**
    - Hint: Use comparison operators.
8. **Check if three sides can form a triangle.**
    - Hint: Use the triangle inequality theorem.
9. **Determine the quadrant of a point in a coordinate system.**
    - Hint: Use nested if statements.
10. **Check if a string contains only digits.**
    - Hint: Use the string method `isdigit()`.
11. **Determine if a person is eligible to vote based on age.**
    - Hint: Use a simple comparison.
12. **Check if a number is within a given range.**
    - Hint: Use comparison operators.
13. **Determine if a password is strong (contains uppercase, lowercase, digits, special characters).**
    - Hint: Use string methods and comparison.
14. **Check if a number is a perfect square.**
    - Hint: Use the square root and check if the result is an integer.
15. **Determine if a given day is a weekday or weekend.**
    - Hint: Use a list of weekdays and weekends.
16. **Check if a year falls within a given range.**
    - Hint: Use comparison operators.
17. **Determine if a string starts and ends with the same character.**
    - Hint: Compare the first and last characters of the string.
18. **Check if a list is sorted in ascending order.**
    - Hint: Compare each element with the next one.
19. **Determine if a number is a multiple of both 3 and 5.**
    - Hint: Use the modulus operator.
20. **Check if a given date is valid (considering leap years).**
    - Hint: Use nested if statements and the rules for leap years.

Loops

1. **Print the first 10 Fibonacci numbers.**
    - Hint: Use a loop to generate the sequence.
2. **Print all prime numbers up to a given number.**
    - Hint: Use a nested loop to check for primality.
3. **Calculate the sum of digits of a number.**
    - Hint: Use a loop to extract and sum digits.
4. **Print a multiplication table for a given number.**
    - Hint: Use a loop to generate the table.
5. **Print the reverse of a given string.**
    - Hint: Use a loop to iterate through the string in reverse.
6. **Calculate the factorial of a number using a loop.**
    - Hint: Use a loop to multiply numbers.
7. **Print all Armstrong numbers up to a given number.**
    - Hint: Use nested loops and the definition of Armstrong numbers.
8. **Calculate the sum of all even numbers in a list.**
    - Hint: Use a loop to iterate through the list and sum even numbers.
9. **Print a right-angled triangle pattern of stars.**
    - Hint: Use nested loops to print the pattern.
10. **Find the largest and smallest elements in a list.**
    - Hint: Use a loop to traverse the list and keep track of the max and min values.
11. **Print all perfect numbers up to a given number.**
    - Hint: Use a loop to check for perfect numbers.
12. **Calculate the sum of squares of the first n natural numbers using a loop.**
    - Hint: Use a loop to sum the squares.
13. **Print a diamond pattern using stars.**
    - Hint: Use nested loops to print the pattern.
14. **Check if a number is a palindrome using a loop.**
    - Hint: Use a loop to compare digits.
15. **Calculate the GCD of two numbers using a loop.**
    - Hint: Use a loop with the Euclidean algorithm.
16. **Print the Pascal's triangle up to a given number of rows.**
    - Hint: Use nested loops and the binomial coefficient formula.
17. **Find the sum of the first n terms of an arithmetic progression.**
    - Hint: Use a loop to sum the terms.
18. **Print all the divisors of a number.**
    - Hint: Use a loop to check for divisors.
19. **Print all pairs of elements in a list.**
    - Hint: Use nested loops to generate pairs.
20. **Print all the permutations of a string.**
    - Hint: Use recursion or the itertools library.

Exceptions

1. **Handle division by zero in a program.**
    - Hint: Use a try-except block.
2. **Handle a file not found error.**
    - Hint: Use a try-except block when opening a file.
3. **Catch and handle a value error when converting input to an integer.**
    - Hint: Use a try-except block around the conversion.
4. **Handle multiple exceptions in a program.**
    - Hint: Use multiple except blocks.
5. **Create a custom exception for invalid inputs.**
    - Hint: Define a new exception class and use raise.
6. **Handle a key error in a dictionary.**
    - Hint: Use a try-except block or the get method.
7. **Catch an index error in a list.**
    - Hint: Use a try-except block.
8. **Handle an import error.**
    - Hint: Use a try-except block around the import statement.
9. **Raise an exception when a condition is not met.**
    - Hint: Use the raise keyword.
10. **Handle a type error in a function.**
    - Hint: Use a try-except block.
11. **Handle an attribute error.**
    - Hint: Use a try-except block.
12. **Handle an exception in a nested try-except block.**
    - Hint: Use a nested try-except structure.
13. **Handle a timeout error in network requests.**
    - Hint: Use a try-except block.
14. **Handle an exception in a loop.**
    - Hint: Use a try-except block inside the loop.
15. **Handle an exception in a function.**
    - Hint: Use a try-except block inside the function.

16. **Handle an exception in a function.**
    - Hint: Use a try-except block inside the function.
17. **Handle a zero division error with a custom message.**
    - Hint: Use a try-except block and print a custom message.
18. **Create a function that raises an exception if the input is negative.**
    - Hint: Use an if statement and raise ValueError.
19. **Handle a file reading exception and print a custom error message.**
    - Hint: Use a try-except block around the file read operation.
20. **Handle an exception in a lambda function.**
    - Hint: Use a try-except block inside a regular function and call the lambda within it.

Libraries

1. **Generate a random number between 1 and 100 using `random` library.**
   - Hint: Use `random.randint(1, 100)`.
2. **Calculate the square root of a number using `math` library.**
   - Hint: Use `math.sqrt()`.
3. **Fetch JSON data from a URL using `requests` library.**
   - Hint: Use `requests.get(url).json()`.
4. **Parse a JSON string using `json` library.**
   - Hint: Use `json.loads()`.
5. **Write a JSON object to a file using `json` library.**
   - Hint: Use `json.dump()`.
6. **Create a DataFrame from a list of dictionaries using `pandas` library.**
   - Hint: Use `pandas.DataFrame()`.
7. **Plot a simple line graph using `matplotlib` library.**
   - Hint: Use `matplotlib.pyplot.plot()` and `show()`.
8. **Perform matrix multiplication using `numpy` library.**
   - Hint: Use `numpy.dot()`.
9. **Calculate the mean and standard deviation of a list using `statistics` library.**
   - Hint: Use `statistics.mean()` and `statistics.stdev()`.
10. **Parse an XML file using `xml.etree.ElementTree` library.**
    - Hint: Use `ElementTree.parse()` and `findall()`.
11. **Create a ZIP file using `zipfile` library.**
    - Hint: Use `zipfile.ZipFile()`.
12. **Send an email using `smtplib` library.**
    - Hint: Use `smtplib.SMTP()`.
13. **Scrape a webpage using `BeautifulSoup` from `bs4` library.**
    - Hint: Use `BeautifulSoup` and `requests.get()`.
14. **Perform date arithmetic using `datetime` library.**
    - Hint: Use `datetime.timedelta()`.
15. **Hash a string using `hashlib` library.**
    - Hint: Use `hashlib.sha256().hexdigest()`.
16. **Generate a QR code using `qrcode` library.**
    - Hint: Use `qrcode.make()`.
17. **Create and manipulate a CSV file using `csv` library.**
    - Hint: Use `csv.reader()` and `csv.writer()`.
18. **Compress data using `zlib` library.**
    - Hint: Use `zlib.compress()` and `zlib.decompress()`.
19. **Schedule a task to run at a specific time using `schedule` library.**
    - Hint: Use `schedule.every().day.at()` and `run_pending()`.
20. **Perform basic image processing using `PIL` (Pillow) library.**
    - Hint: Use `PIL.Image.open()` and `Image.filter()`.

File I/O

1. **Read a text file and print its contents.**
    - Hint: Use `open()` and `read()`.
2. **Write a list of strings to a text file.**
    - Hint: Use `open()` and `writelines()`.
3. **Append a string to a text file.**
    - Hint: Use `open()` with mode 'a'.
4. **Read a CSV file and print its contents.**
    - Hint: Use `csv.reader()`.
5. **Write a list of dictionaries to a CSV file.**
    - Hint: Use `csv.DictWriter()`.
6. **Read a JSON file and convert it to a dictionary.**
    - Hint: Use `json.load()`.
7. **Write a dictionary to a JSON file.**
    - Hint: Use `json.dump()`.
8. **Read an image file and display its properties.**
    - Hint: Use `PIL.Image.open()`.
9. **Write binary data to a file.**
    - Hint: Use `open()` with mode 'wb'.
10. **Read binary data from a file.**
    - Hint: Use `open()` with mode 'rb'.
11. **Copy the contents of one file to another.**
    - Hint: Use `shutil.copyfile()`.
12. **Read a large file line by line.**
    - Hint: Use `open()` with a loop to iterate through lines.
13. **Write logs to a file using `logging` library.**
    - Hint: Use `logging.basicConfig()`.
14. **Read and parse an XML file.**
    - Hint: Use `xml.etree.ElementTree.parse()`.
15. **Count the number of lines, words, and characters in a text file.**
    - Hint: Use `open()` and a loop to count.
16. **Create a new directory.**
    - Hint: Use `os.mkdir()`.
17. **List all files in a directory.**
    - Hint: Use `os.listdir()`.
18. **Rename a file.**
    - Hint: Use `os.rename()`.
19. **Delete a file.**
    - Hint: Use `os.remove()`.
20. **Compress a file using `gzip` library.**
    - Hint: Use `gzip.open()`.

Regular Expressions

1. **Validate an email address.**
    - Hint: Use `re.match()` with an email pattern.
2. **Find all dates in a text.**
    - Hint: Use `re.findall()` with a date pattern.
3. **Replace all occurrences of a word in a string.**
    - Hint: Use `re.sub()`.
4. **Extract all URLs from a text.**
    - Hint: Use `re.findall()` with a URL pattern.
5. **Split a string by commas.**
    - Hint: Use `re.split()`.
6. **Check if a string contains only digits.**
    - Hint: Use `re.match()` with a digit pattern.
7. **Find all words starting with a specific letter.**
    - Hint: Use `re.findall()` with a word pattern.
8. **Replace multiple spaces with a single space.**
    - Hint: Use `re.sub()`.
9. **Extract the domain from an email address.**
    - Hint: Use `re.search()`.
10. **Find all phone numbers in a text.**
    - Hint: Use `re.findall()` with a phone number pattern.
11. **Check if a string is a valid IPv4 address.**