Python Dictionary and Control Statements: Practical Programming Examples

1. Dictionary Manipulation: Write a Python function that takes two dictionaries as input and returns a new dictionary containing all key-value pairs from both dictionaries. If a key exists in both dictionaries, the value from the second dictionary should be used.

2. Nested Dictionary Access: Given a nested dictionary representing a file system structure, write a function that takes a path as a string (e.g., "/home/user/documents") and returns the corresponding nested dictionary object. Handle cases where the path doesn't exist.

3. Dictionary Comprehension: Create a dictionary using dictionary comprehension that contains the squares of all even numbers from 1 to 20 as values, with the numbers themselves as keys.

4. Control Flow in Dictionary Operations: Write a function that takes a dictionary of student scores (where keys are student names and values are lists of scores) and returns a new dictionary with the same keys, but the values are the letter grades based on the average score. Use the following grading scale: A: 90-100, B: 80-89, C: 70-79, D: 60-69, F: below 60

5. Conditional Dictionary Update: Write a function that updates a dictionary of stock prices. The function should take the current stock dictionary, a stock name, and a new price. It should only update the price if it's higher than the current price for that stock.

6. Loop and Dictionary: Write a program that repeatedly asks the user for a word and its definition, storing them in a dictionary. Use a loop to continue this process until the user enters 'quit'. Then print out all the words and their definitions.

7. Exception Handling with Dictionaries: Create a function that safely retrieves a value from a nested dictionary given a list of keys. Use exception handling to deal with cases where a key doesn't exist at any level.

8. Dictionary Sorting: Write a function that takes a dictionary and returns a list of tuples (key, value) sorted by value in descending order. If two values are the same, sort by key in ascending order.

9. Conditional Dictionary Filtering: Given a dictionary of products and their prices, write a function that returns a new dictionary containing only the products that are within a specified price range.

10. Dictionary-based Menu System: Create a menu-driven program where the menu options are stored in a dictionary. The keys should be the option numbers, and the values should be tuples containing the option text and the function to call for that option. Use a loop and control statements to handle user input and execute the appropriate function.