Random Password Generator

1. Importing Necessary Modules

```
import random
import string
```

- `random`: This module is used to generate random numbers and select random elements.
- `string`: This module contains a collection of string constants like `ascii_lowercase`, `ascii_uppercase`, `digits`, and `punctuation`.

2. Password Generator Function

```
def password_generator(length, complexity):
    char_sets = {
        1: string.ascii_lowercase + string.ascii_uppercase,
        2: string.ascii_lowercase + string.ascii_uppercase + string.digits,
        3: string.ascii_lowercase + string.ascii_uppercase + string.digits
+ string.punctuation}
    characters = char_sets[complexity]
    while True:
        pwd = ''.join(random.choice(characters)
        for i in range(length))
        yield pwd
```

- `char_sets`: A dictionary mapping complexity levels (1, 2, and 3) to different sets of characters:
  - Complexity 1: Lowercase and uppercase letters.
  - Complexity 2: Lowercase and uppercase letters and digits.
  - Complexity 3: Lowercase and uppercase letters, digits, and punctuation.
- `characters`: Selects the appropriate set of characters based on the given complexity.
- `while True`: An infinite loop to keep generating passwords.
- `pwd = ''.join(random.choice(characters) for i in range(length))`: Generates a password of the specified length by randomly choosing characters from the selected set.
- `yield pwd`: Returns the generated password.

3. Main Function

```
def main():
    while True:
        opt = input("Press 0 to generate Password (or) OFF to switch off
generator: ")
        if opt == 'OFF':
            break
        elif opt == "0":
            try:
                length = int(input("Enter the password length: "))
                complexity = int(input("Enter the complexity (1-3): "))
                if complexity not in range(1, 4):
```

```
                    print("** Enter valid complexity value (1-3) **")
                    continue
                p = password_generator(length, complexity)
                pwd = next(p)
                print("Generated Password: ", pwd)
            except ValueError:
                print("** Enter valid integer values for length and
complexity **")
        else:
            print("*** Enter valid inputs ***")
main()
```

- `while True`: An infinite loop to keep the program running until the user decides to exit.
- `opt = input(...)`: Prompts the user to either generate a password (`0`) or exit the program (`OFF`).
- `if opt == 'OFF': break`: Exits the loop and ends the program if the user inputs `OFF`.
- `elif opt == "0":`: Proceeds to generate a password if the user inputs `0`.
- `else: print("*** Enter valid inputs ***")`: Prints an error message if the input is not `0` or `OFF`.

The code creates a user-interactive console application that generates passwords based on user-specified length and complexity. It uses a generator function to produce the passwords and handles various edge cases and invalid inputs gracefully.