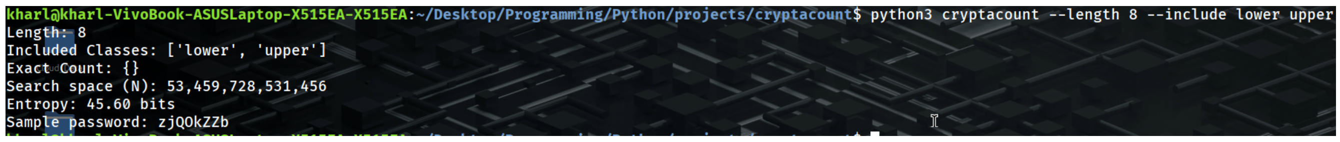
Project Title: Cryptacount

Name: Bugarin, Kharl Denzell D.

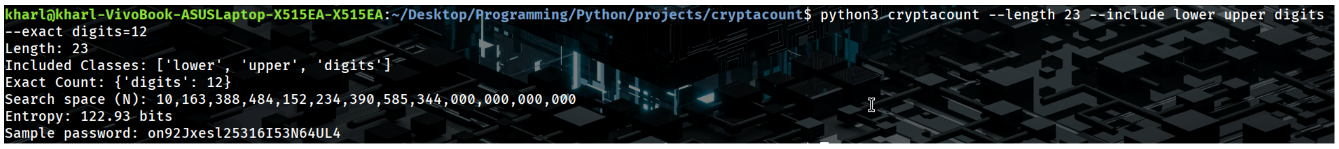
Project Description:

CryptaCount is a **Python-based program** designed to simulate and calculate the combinatorial possibilities of password creation. It serves two primary functions: a **calculator** of password strength and a **generator** of valid password samples. The core objective is to demonstrate the powerful, real-world application of discrete mathematics principles, specifically the exponential growth of possibilities in the context of computer security and entropy analysis.

Screenshots:



```
kharl@kharl-VivoBook-ASUSLaptop-X515EA-X515EA:~/Desktop/Programming/Python/projects/cryptacount$ python3 cryptacount --length 8 --include lower upper
Length: 8
Included Classes: ['lower', 'upper']
Exact Count: {}
Search space (N): 53,459,728,531,456
Entropy: 45.60 bits
Sample password: zjQOkZZb
```



```
kharl@kharl-VivoBook-ASUSLaptop-X515EA-X515EA:~/Desktop/Programming/Python/projects/cryptacount$ python3 cryptacount --length 23 --include lower upper digits
--exact digits=12
Length: 23
Included Classes: ['lower', 'upper', 'digits']
Exact Count: {'digits': 12}
Search space (N): 10,163,388,484,152,234,390,585,344,000,000,000,000
Entropy: 122.93 bits
Sample password: on92Jxesl25316I53N64UL4
```

Explanation of Formulas/Rules Implemented:

The core challenge is solving the "placement problem": If a password of length L must contain exactly k_1 items of character type 1 (e.g., 2 digits) and k_2 items of type 2, how many ways can these required characters be arranged?

This is solved by calculating the number of unique structural arrangements using the **Multinomial Coefficient**, which is a generalized form of the **Combination** formula. This formula determines the number of ways to partition the total password length (L) into the required counts (k_i) and the remaining count (r).

Placement Ways = L! / k_1! * k_2! * * * r!

- **Implementation:** This is handled by the multinomial_coefficient function in the program.
- **Significance:** This result counts the number of ways to *arrange the slots* (e.g., 28 ways to place 2 digits in 8 slots) before filling them with characters.

The Fundamental Driver: Rules of Sum & Product

The Rule of Product is applied to find the final search space (N) by multiplying three independent counting factors, each derived from the password criteria:

Total Search Space (N) = (Placement Ways) times (Fill Choices for Required Chars) times (Fill Choices for Remaining Chars)

This process ensures that the result from the P&C logic is correctly combined with the character selection logic.

Final Blended Formula

The full search space calculation implements the Rule of Product by multiplying the three factors below:

N = (L! / k! * r!) * (C_req)^k * (C_other}^r

Factor	***Concept***	***Explanation***
(L! / k! * r!)	***Permutations & Combinations*** *(Placement)*	The number of ways to arrange the required items (k) and the remaining items (r) within the total length (L).
(C_req)^k	***Rule of Product*** *(Required Fill)*	The number of ways to fill the k required slots using the C_req characters available in that set (e.g., 10^2 for two digits).
$(C_other)^r	***Rule of Product*** *(Remaining Fill)*	The number of ways to fill the r remaining slots using the C_other characters available in the pool of all other allowed sets.

Conclusion

The CryptaCount project successfully simulated a complex real-world security scenario and demonstrated the direct, powerful role of discrete mathematics in cryptography. By correctly implementing the logic of Permutations & Combinations and the Rules of Sum & Product, the program achieved its core goal: calculating the true entropy of a password based on precise constraints.

The project proved that password strength is not linear; it is exponential. The program's output, with search spaces reaching into the trillions, visually and computationally confirms that even small changes to a password's length or character diversity (which translate directly to the L and C variables in the formulas) result in massive, exponential security gains. This illustrates the fundamental importance of combinatorial analysis in risk assessment and modern password policy.

Reflection

This project was successful in translating the theoretical concepts of Permutations & Combinations and the Rules of Sum & Product into a highly relevant, functional security tool. The core learning outcome was demonstrating that complexity in passwords is not linear but exponential, with small changes in length or character set causing trillions of additional possibilities due to the compounding effect of the Rule of Product. The main technical challenge was ensuring the mathematical accuracy of the final calculation (N), which required careful implementation of the advanced Multinomial Coefficient logic and the use of Python's large integer handling to correctly process the massive search space figures without error, thereby fulfilling all requirements for concept mastery and technical quality.