# Password Strength Checker

# Introduction:

PASSWORD STRENGTH CHECKER IS A TOOL DESIGNED TO EVALUATE THE SECURITY OF A PASSWORD

• LENGTH

· CHARACTER VARIETY

· COMMON PATTERNS

· ENTROPY

# Mathematical Foundations of Password Strength Checker:

- Combinatorics

  N = c^L

- Entropy

  H = log (c)  base 2

  - Number theory
  - Pattern matching dictionary lookup

# Password Strength Checker and Logical Proof:

This is how password strength checker works:

Let, P be a Password

U(P) = Password has uppercase letter

L(P) = Password has lowercase letter

D(P) = Password has digit

S(P) = Password has special character

R(P) = Password is strong

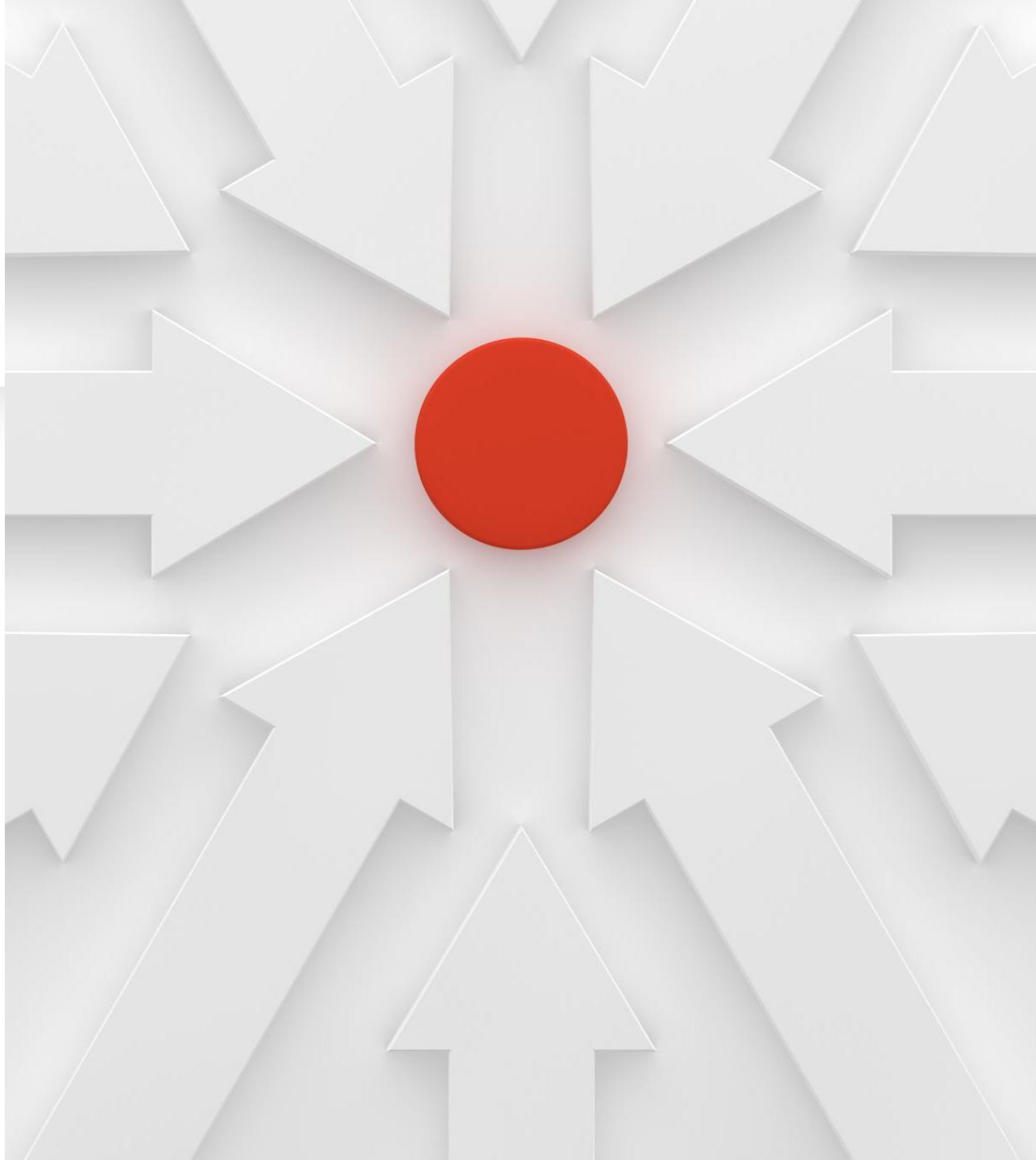
$(U(P) \land L(P) \land D(P) \land S(P)) \rightarrow R(P)$

- Proof by cases
- Case 1 (Too Short)
- Case 2 (Lacks Complexity)
- Case 3 (Predictable Patterns)
- Case 4 (Repeated or Sequential Characters)

A password P is weak if it meets any of the case

# Password Strength Checker and Set Theory:

- Cardinality (Should be greater)
- Union (Should be greater)
- Intersection (Should be ∅)



# Password Strength Checker and Relations:

- Attribute Relations:
  - Length vs. Character Diversity
  - Predictability vs. Length

# Password strength checker and Functions:

f(password) = strength

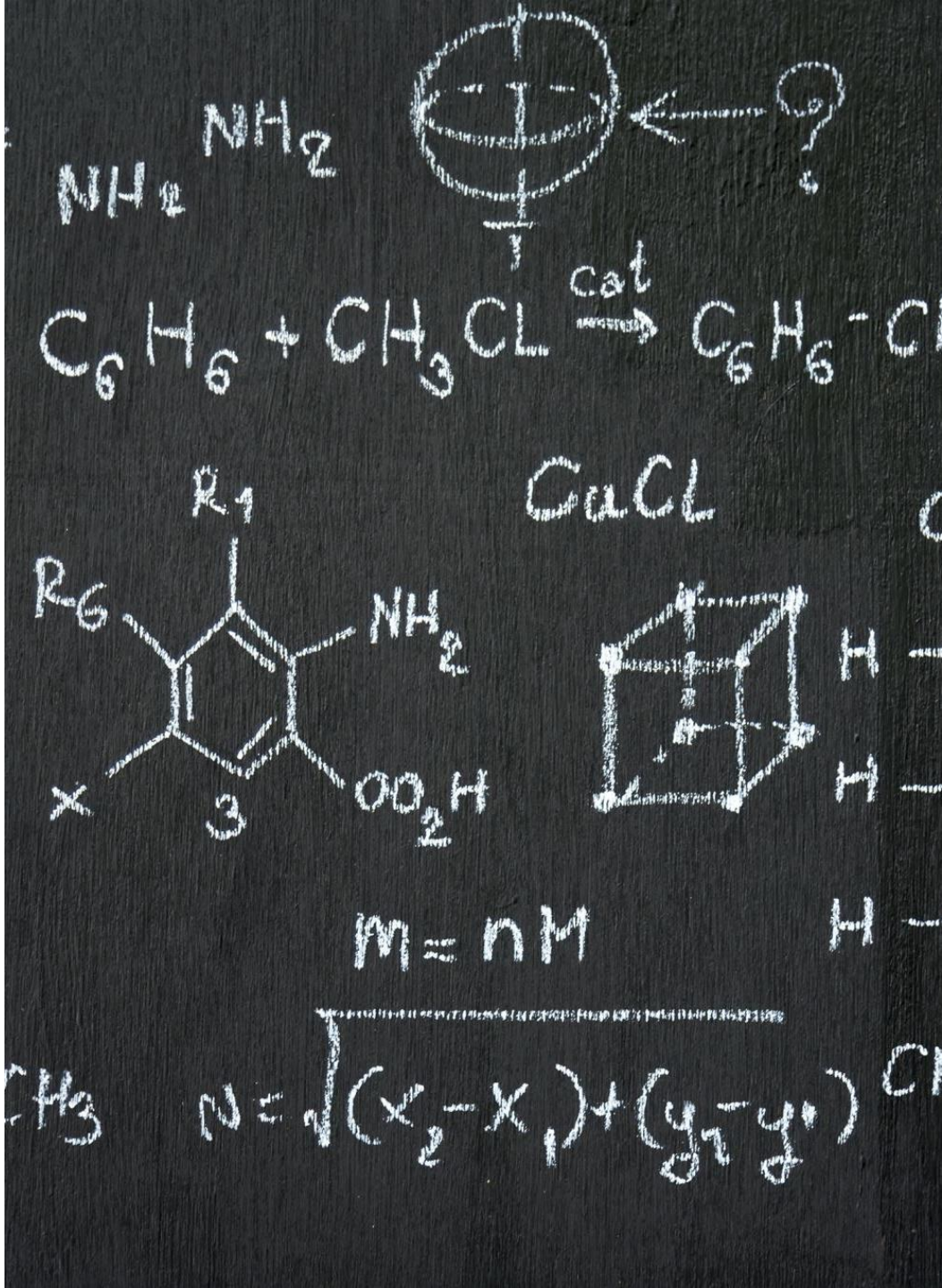
Domain : string of characters {A,B,….,Z,a,b,….,z,0,1,….,9,` ,~,[ , ; , : , …..}

Range: strength of password {weak,moderate,strong}

- Inverse relation:

We can use inverse function for suggestion of strong password

f^-1(strong) = password



# Password Strength Checker and Probability:

- Probability of Successful Guessing

  P = 1/N  (N =c^L)

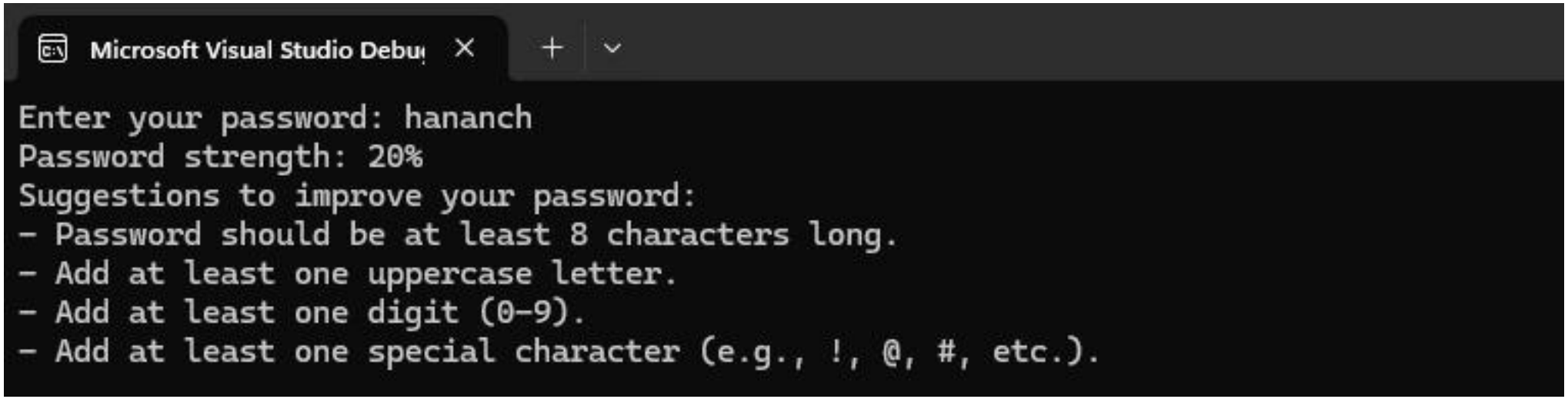
# Computer Program for Password Strength Checker:

```
bool isUppercase(const string& pass)
 {
    for (char ch : pass) {
        if (isupper(ch)) return true;
    }
  return false;
}
bool isLowercase(const string& pass)
 {
    for (char ch : pass) {
        if (islower(ch)) return true;
    }
    return false;
}
```

```
bool isDigit(const string& pass) {
    for (char ch : pass) {
        if (isdigit(ch)) return true;
    }
    return false;
}
bool isSpecialChar(const string& pass) {
    string specialChars = "!@#$%^&*()-
  _=+[{]}|;:'\",<.>/?`~";
    for (char ch : pass) {
        if (specialChars.find(ch) !=
  string::npos) return true;
    }
    return false;
}
```

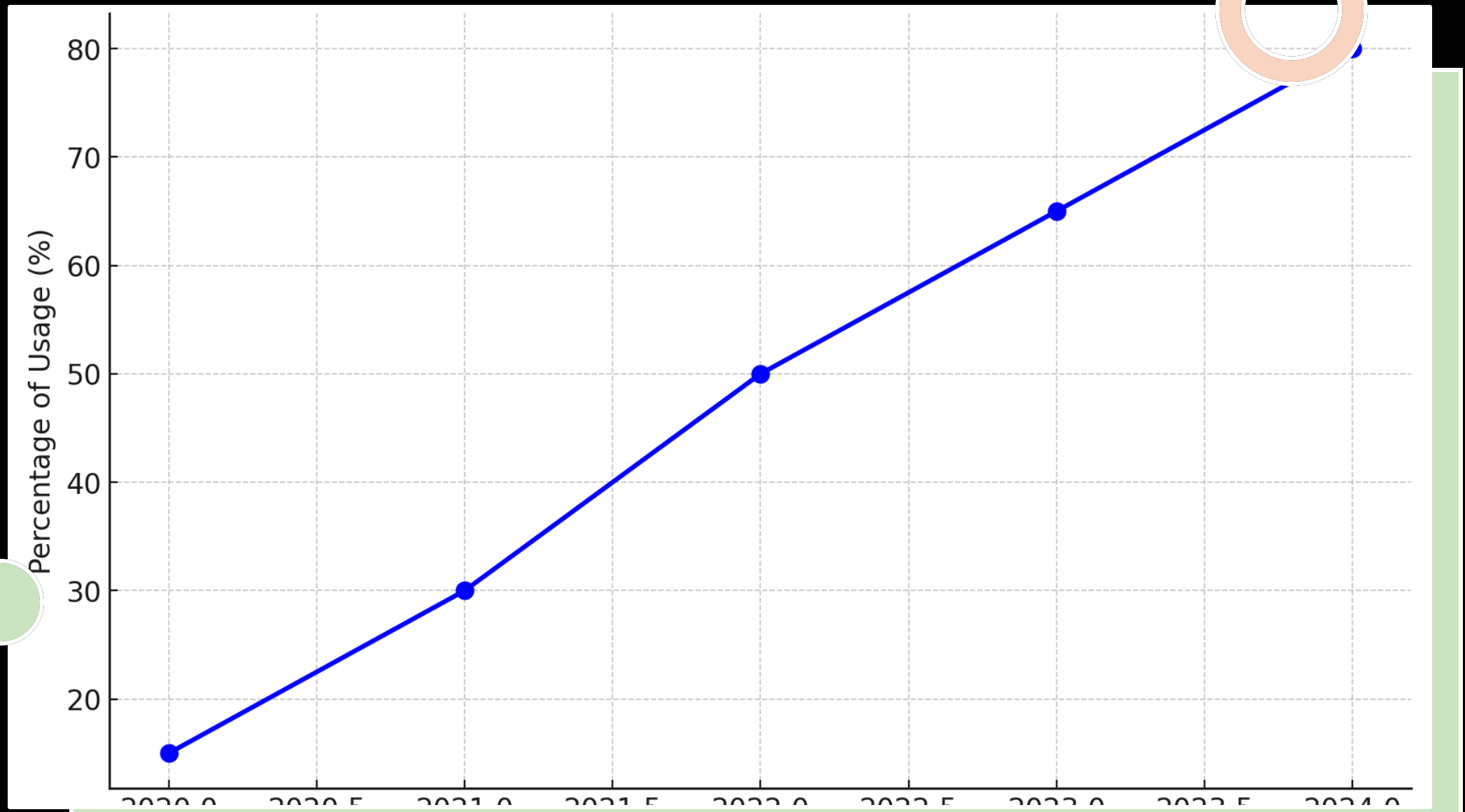
```
int PasswordStrength(const string& pass, string&
  feedback) {

    int score = 0;

    int totalCriteria = 5;

    if (pass.length() >= 8) score += 1;

    else feedback += "- Password should be at least 8
  characters long.\n";

    if (isUppercase(pass)) score += 1;

    else feedback += "- Add at least one uppercase
  letter.\n";

    if (isLowercase(pass)) score += 1;

    else feedback += "- Add at least one lowercase
  letter.\n";

    if (isDigit(pass)) score += 1;

    else feedback += "- Add at least one digit (0-9).\n";

    if (isSpecialChar(pass)) score += 1;

    else feedback += "- Add at least one special
  character (e.g., !, @, #, etc.).\n";

    int percentage = (score * 100) / totalCriteria;

    return percentage;

}
```

```
int main() {
    string password;
    string feedback = "";
    cout << "Enter your password: ";
    cin >> password;
    int strengthPercentage =
  PasswordStrength(password, feedback);
    cout << "Password strength: " <<
  strengthPercentage << "%" << endl;
    if (strengthPercentage < 100) cout <<
  "Suggestions to improve your password:\n"
  << feedback;
    else cout << "Your password is strong!"
  << endl;
    return 0;
}
```


Microsoft Visual Studio Debu
Enter your password: hananch
Password strength: 20%
Suggestions to improve your password:
- Password should be at least 8 characters long.
- Add at least one uppercase letter.
- Add at least one digit (0-9).
- Add at least one special character (e.g., !, @, #, etc.).

# Real World Applications of Password Strength Checker:

- Cryptocurrency Security
- Protecting Digital Assets
- Cloud Storage Services:
- Online Banking & Financial Services:
- Social Media Platforms:
- Automated Home Lockdown:


Percentage of Usage (%)
80
70
60
50
40
30
20

# Conclusion:

Password Strength Checker is not just a technical tool but a critical component of a comprehensive security strategy