1.حل سودوکو به روش بازگشتی(Backtracking)

مباحث کلیدی :آرایه‌ها، توابع، حلقه‌ها، شرط‌ها، الگوریتم بازگشت
شرح تمرین:
یک برنامه بنویسید که جدول سودوکو (مثلاً اندازه 9×9) را از کاربر یا فایل ورودی دریافت کند و با استفاده از الگوریتم Backtrackingآن را حل کند.

2.هوش مصنوعی در بازی سنگ-کاغذ-قیچی (با یادگیری ساده)

مباحث کلیدی :رشته‌ها(strings) ، آرایه‌ها، حلقه‌ها، شرط‌ها، توابع
شرح تمرین:
بازی "سنگ-کاغذ-قیچی" را پیاده‌سازی کنید تا کاربر بتواند با کامپیوتر بازی کند.
سپس یک مکانیزم یادگیری ساده اضافه کنید که با ذخیره و تحلیل تاریخچه‌ی حرکات کاربر، حرکت بعدی او را پیش‌بینی کند.
مثلاً اگر کاربر در سه نوبت گذشته "سنگ" انتخاب کرده باشد، برنامه احتمال دهد که دوباره "سنگ" انتخاب می‌شود و در نتیجه "کاغذ" را انتخاب کند تا برنده شود.

3.بررسی تعادل پرانتزها(Balanced Parentheses Checker)

مباحث کلیدی :رشته‌ها، ساختار داده‌ی پشته(Stack) ، حلقه‌ها، شرط‌ها
شرح تمرین:
برنامه‌ای بنویسید که یک رشته شامل پرانتزهای [] ,{} ,()را دریافت کند و بررسی کند که آیا پرانتزها به درستی باز و بسته شده‌اند یا نه.
برای این کار استفاده از Stack پیشنهاد می‌شود، چرا که ساختاری مناسب برای بررسی تطابق ورودی‌های لایه‌دار است.
مثال:

- **{()[()()]{}} → متعادل ✅**
- **[(]) → نامتعادل ❌**

4.شمارش تعداد جزایر در ماتریس(Number of Islands)

مباحث کلیدی :آرایه‌ها، بازگشت یاBFS/DFS ، شرط‌ها، توابع
شرح تمرین:
در یک ماتریس m × nشامل 0 و 1، عدد 1 نشان‌دهنده خشکی و عدد 0 نشان‌دهنده آب است.
هر جزیره مجموعه‌ای از 1هاست که به‌صورت افقی یا عمودی به هم متصل هستند.

هدف برنامه این است که تعداد جزایر موجود را بشمارید.

5.مسئله چینش بهینه آنتن‌های مخابراتی(Cell Tower Placement Problem)

مباحث کلیدی :بهینه‌سازی ترکیبیاتی، الگوریتم‌های جست‌وجو
شرح تمرین:
در این مسئله قصد داریم موقعیت‌های بهینه برای قرار دادن آنتن‌های مخابراتی مثلاً 5G را در یک ناحیه مشخص تعیین کنیم.
اهداف و محدودیت‌ها:

1. **حداکثر پوشش‌دهی کاربران.(ideally 100٪)**
2. **کمینه‌سازی تعداد آنتن‌ها یا هزینه نصب آن‌ها**

صورت مسئله: پیاده‌سازی سیستم رمزنگاری و رمزگشایی داده‌های متنی

شرح مسئله:

برنامه‌ای طراحی کنید که یک فایل متنی حاوی اطلاعات حساس را رمزنگاری کرده و امکان رمزگشایی آن را فراهم کند. برنامه باید فایل متنی را بخواند، محتوای آن را رمزنگاری کند، خروجی رمزنگاری‌شده را در یک فایل جدید ذخیره کند و سپس بتواند فایل رمزنگاری‌شده را با استفاده از کلید صحیح رمزگشایی کرده و محتوای اصلی را بازتولید کند.

ورودی‌ها:

- یک فایل متنی (مثلاً `input.txt`) حاوی داده‌های حساس.

- یک کلید رمزنگاری (تولیدشده به‌صورت امن یا واردشده توسط کاربر).

خروجی‌ها:

- فایل رمزنگاری‌شده (مثلاً `encrypted.bin`) حاوی داده‌های رمز شده.

- فایل رمزگشایی‌شده (مثلاً `decrypted.txt`) که محتوای اصلی را بازتولید می‌کند.

الزامات:

1. رمزنگاری باید با یک روش امن و استاندارد انجام شود، به‌گونه‌ای که بدون کلید صحیح امکان رمزگشایی وجود نداشته باشد.

2. کلید رمزنگاری باید به‌صورت امن تولید یا مدیریت شود.

3. برنامه باید شامل دو تابع اصلی باشد:

   - تابع رمزنگاری: فایل متنی را خوانده، رمزنگاری کرده و خروجی را در فایل جدید ذخیره کند.

   - تابع رمزگشایی: فایل رمزنگاری‌شده را خوانده و با استفاده از کلید صحیح، محتوای اصلی را بازتولید کند.

4. (اختیاری) از روشی برای اعتبارسنجی یکپارچگی داده‌ها استفاده کنید.

محدودیت‌ها:

- کلید باید به صورت امن مدیریت شود .

- برنامه باید بتواند فایل‌های بزرگ را به‌صورت کارآمد پردازش کند.

مثال:

- فایل ورودی:input.txt با محتوای "This is a secret message."

- کلید: یک رشته امن تولیدشده.

- خروجی رمزنگاری: فایل encrypted.bin حاوی داده‌های رمز شده.

- خروجی رمزگشایی: فایلdecrypted.txt با محتوای اصلی "This is a secret message."

تحویل‌دادنی‌ها:

- کد برنامه با توضیحات کافی.

- مستندات مختصر شامل نحوه اجرا و مدیریت کلید(README.md).