پروژه ساختمان داده

مدیریت کتابخانه

ابراهیم خدادادی

دی 1404

هدف این پروژه، طراحی و پیاده‌سازی یک سیستم مدیریت کتابخانه ساده به زبان C++ است که امکان انجام عملیات اصلی یک کتابخانه واقعی از جمله افزودن، حذف، جستجو، مرتب‌سازی کتاب‌ها، مدیریت درخواست‌های امانت و نگهداری تاریخچه عملیات را فراهم کند.

این برنامه با تمرکز بر مفاهیم مهم درس ساختمان داده‌ها پیاده‌سازی شده و از ساختمان‌های داده پویا مانند **لیست پیوندی دوطرفه**، **صف** و **پشته** استفاده کرده است تا کارایی، انعطاف‌پذیری و مدیریت صحیح حافظه را تضمین کند. برنامه به صورت شئ‌گرا (Object-Oriented) طراحی شده و دارای منوی تعاملی است که کاربر می‌تواند به راحتی با سیستم کار کند.

ساختار کلی برنامه

برنامه از چندین کلاس اصلی تشکیل شده است:

- **کلاس :Book**نمایانگر یک کتاب با ویژگی‌های کد یکتا، عنوان، نویسنده، سال انتشار و وضعیت (موجود / امانت داده شده).
- **کلاس :Library** مدیریت مجموعه کتاب‌ها با استفاده از **لیست پیوندی دوطرفه.(Doubly Linked List)**
- **کلاس :LoanQueue** مدیریت درخواست‌های امانت کتاب با استفاده از **صف (Queue)** پیاده‌سازی شده با لیست پیوندی.
- **کلاس :HistoryStack** نگهداری تاریخچه آخرین عملیات انجام‌شده با استفاده از **پشته (Stack)** با ظرفیت محدود (حداکثر ۵ عملیات).

بخش‌های اصلی برنامه و نحوه عملکرد

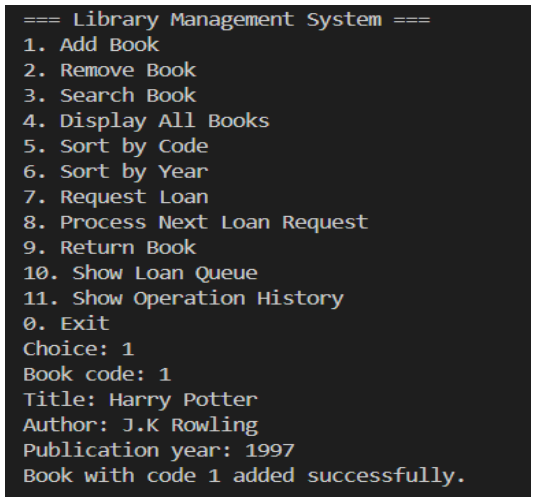
1. **ذخیره‌سازی و مدیریت کتاب‌ها** تمام کتاب‌ها در یک لیست پیوندی دوطرفه نگهداری می‌شوند. این ساختار امکان افزودن و حذف سریع کتاب‌ها را فراهم می‌کند و اندازه آن به صورت پویا تغییر می‌کند (برخلاف آرایه‌های ثابت) .عملیات‌های اصلی شامل:
    - افزودن کتاب جدید
    - حذف کتاب بر اساس کد
    - جستجوی کتاب بر اساس کد یکتا
    - نمایش تمام کتاب‌ها
2. **مرتب‌سازی کتاب‌ها** برنامه امکان مرتب‌سازی کتاب‌ها را بر اساس دو معیار فراهم کرده است:
    - مرتب‌سازی بر اساس کد کتاب (صعودی)
    - مرتب‌سازی بر اساس سال انتشار (صعودی) الگوریتم مرتب‌سازی بر روی لیست پیوندی اعمال می‌شود و از روش ساده اما قابل فهم) مانند (Bubble Sort) استفاده شده است.
3. **مدیریت امانت کتاب** درخواست‌های امانت کتاب در یک **صف (Queue)** به ترتیب ورود (FIFO) ذخیره می‌شوند .وقتی کاربر درخواست پردازش بعدی را انتخاب می‌کند:
    - اولین درخواست از صف برداشته می‌شود.
    - اگر کتاب مورد نظر موجود باشد، وضعیت آن به "امانت داده شده" تغییر می‌کند.
    - در غیر این صورت، درخواست دوباره به انتهای صف بازگردانده می‌شود تا بعداً بررسی شود . بازگشت کتاب نیز پشتیبانی می‌شود و وضعیت کتاب به "موجود" تغییر می‌یابد.

استفاده از صف با لیست پیوندی باعث شده که هیچ محدودیتی در تعداد درخواست‌ها وجود نداشته باشد و مشکل سرریز (overflow)صف‌های آرایه‌ای حل شود.

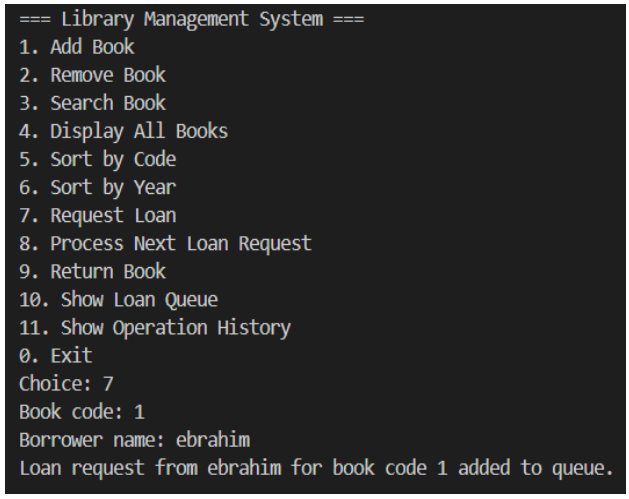
4. **تاریخچه عملیات (Operation History)** برای برآورده کردن نیاز نمایش آخرین عملیات‌ها، از یک **پشته (Stack)**با ظرفیت ۵ استفاده شده است .پس از انجام اکثر عملیات‌ها (مانند افزودن، حذف، امانت، مرتب‌سازی و ...)، توضیح مختصری از آن عملیات در پشته ذخیره می‌شود .وقتی ظرفیت پر شود، قدیمی‌ترین عملیات به طور خودکار حذف و عملیات جدید جایگزین می‌شود .کاربر می‌تواند در هر لحظه آخرین عملیات‌های انجام‌شده را به ترتیب از جدید به قدیم مشاهده کند.

نمونه خروجی برنامه :

1) افزودن یک کتاب



2) ثبت یک درخواست امانت



3) مرحله نهایی و امانت دادن کتاب براساس صف درخواست.