1- پشته ساختمان داده‌ای است که قانون آخرین ورودی اولین خروجی را پیاده‌سازی می‌کند. در زیر تعاریف دو کلاس آورده شده است. Prototype های متدها در کلاس مشخص است. می‌خواهیم پشته‌ای را با استفاده از لیست پیوندی پیاده‌سازی کنیم. برای متدها بدنه بنویسید به گونه‌ای که در برنامه‌ی اصلی خروجی همانند زیر داشته باشیم. فرض بر این است که داده‌ها از جنس عدد صحیح هستند.

```
class Node
{
public:
	Node(const int& d,Node* nxt=0):data(d),next(nxt){}
	int data;
	Node* next;
};
class Stack
{
public:
	Stack();
	Stack(const Stack&);
	~Stack();
	int size() const;
	bool empty() const;
	int& top();
	void push(const int&);
	void pop();
private:
	Node* _top;
	int _size;
};
```

2- پیاده‌سازی کلاس string

کلاسی با نام string ایجاد کنید که دارای فیلد و متدهای زیر باشد

فیلدها

- یک فیلد برای ذخیره کردن رشته
- یک فیلد برای ذخیره کردن طول رشته

اگر فیلد دیگری نیاز دارید خودتان تعریف کنید

فیلدها به صورت خصوصی تعریف شوند

متدها

- سازنده و سازنده‌ی کپی

- دو overload مختلف برای سازنده (بدون ورودی و ورودی char*)
- اپراتورهای درج در جریان و استخراج از جریان
- تابع append را به گونه‌ای بنویسید که یک string را به شی جاری از جنس string متصل نماید.
- اپراتور =+ را به کونه‌ای بنویسید که همانند append عمل کند
- اپراتور = را برای انتساب یک شی از جنس string به شی دیگر خودتان بنویسید.
- تابع length به گونه‌ای که طول رشته را بازگرداند.

در برنامه‌ی اصلی با گرفتن اشیا از کلاس مورد نظر پیاده‌سازی خود را تست کنید.

استفاده از cstring مجاز نمی باشد.

3- آرایه‌ی انعطاف‌پذیر:

آرایه ها به لحاظ بازده دارای مشکلاتی هستند. برای این که مشکلات آرایه‌ها همانند درج در میانه‌ی آرایه یا حذف از مکان میانی را به گونه‌ای مدیریت کنیم از دو ساختار ارائه شده در پیوست استفاده خواهیم کرد.

فایل‌های سرآیند نوشته شده است، برای DynamicArray توابعی همانند زیر وجود دارند:

تابع append: عنصری را دریافت می‌کند و به انتهای آرایه اضافه می‌کند.

تابع remove: با دریافت اندیس خانه‌ی مورد نظر را حذف می‌کند.

تابع resize : می‌خواهیم آرایه‌ی مورد نظر هیچ‌گاه پر نشود و هنگام خالی شدن نیز فضاهای اضافی تخلیه شود. در هنگام فراخوانی این تابع بسته به شرایط می خواهیم به دو گونه عمل کند:

- در زمانی که آرایه پر است، اندازه‌ی آرایه دوبرابر گردد.
- اگر تعداد خانه‌های پر از نصف کل خانه‌ها کمتر است اندازه‌ی آرایه را نصف کند.

.....

توابع را مطالعه کنید و کاملا بر عملکرد آنها مسلط شوید.

بعضی از توابع تنها دارای `prototype` هستند که می باید بدنه‌ی متناظر برای آنها بنویسید. تابع `clear` می‌باید تمام عناصر را تخلیه و تعداد را به صفر برساند.

پیاده‌سازی دیگری برای آرایه‌ی انعطاف پذیر با نام `LinkedArray` در پیوست وجود دارد که آرایه را با لیست پیوندی پیاده‌سازی کرده است. برای توابع کلاس مذکور نیز مطالعه‌ی لازم صورت داده هر جا نیاز به تکمیل بدنه وجود دارد، فرآیند را صورت دهید.

در آینده از این دو کلاس استفاده‌های بسیاری خواهید کرد.

به دلیل وابستگی بین تمرینات از این پس وقت لازم برای حل مسایل اختصاص دهید.