ساختمان داده

تمرین اختیاری اول

بخش پیاده‌سازی

برای پیاده‌سازی این پروژه تنها مجاز به استفاده از کتابخانه‌های زیر می‌باشید:

Functions and Classes	Header File
همه توابع و کلاس‌ها	#include <iostream>
assert(predicate);	#include <cassert>

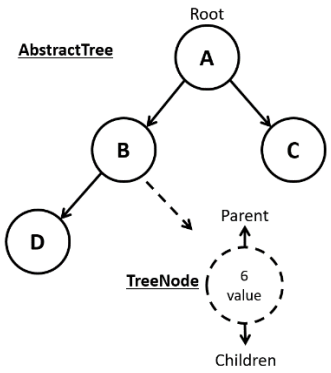
std::min(T, T);
std::max(T, T);
std::swap(T&, T&);
	#include <algorithm>

همه توابع	#include <cmath>
همه ثابت‌ها	#include <cfloat>
همه ثابت‌ها	#include <climits>
کلاس vector	#include <vector>

برای هر تابع کامنت‌گذاری کرده و موارد زیر را توضیح دهید:

- عملکرد تابع
- کاربرد هر پارامتر
- فرضیات تابع
- خطاهایی که ممکن است رخ دهد و نحوه مدیریت‌شان

نام کلاس شما باید دقیقا AbstractTree بوده و در فایل abstract_tree.h پیاده‌سازی شود. توابع موردنیاز درخت انتزاعی در ادامه توضیح داده شده‌اند. فایل امضای توابع ضمیمه تمرین گردیده است.



شکل ۱: نمایش یک نمونه درخت انتزاعی (AbstractTree) و ساختار هر گره (TreeNode)

UML Class Diagram

AbstractTree
- root:TreeNode - maxDegree:Integer
+ create(in maxDegree: Integer):AbstractTree + create(in abt: AbstractTree):AbstractTree + size():Integer + empty():Boolean + height():Integer + find(in value: Integer):Boolean + count(in value: Integer):Integer + push(in value: Integer) + push(in values: Integer, in count: Integer) + pop():Integer + printBFS() + printDFS() + destroy()

توضیحات فیلدها و توابع

فیلد/تابع	توضیحات
root	اشاره‌گر به ریشه درخت
maxDegree	بیشترین درجه‌ای که یک گره در درخت می‌تواند داشته باشد
create()	سازنده پیش‌فرض که یک درخت خالی ایجاد می‌کند. $O(1)$
create(&AbstractTree)	یک درخت دریافت کرده و همه عناصر آن را در خود کپی می‌کند. $O(n)$ تا $O(n \lg n)$
size()	تعداد عناصر موجود در درخت را برمی‌گرداند. $O(1)$ تا $O(n)$
empty()	در صورتی که درخت خالی باشد، مقدار true و درغیراینصورت false برمیگرداند. $O(1)$
height()	ارتفاع درخت را برمی‌گرداند. $O(1)$ تا $O(n)$
find(Integer)	در صورت وجود گره‌ای که مقدارش با مقدار پارامتر ورودی برابر باشد، true برمی‌گرداند. در صورت یافت نشدن مقدار مربوطه، مقدار false برگردانده می‌شود. $O(n)$
count(Integer)	تعداد عناصری که مقدارشان با مقدار پارامتر ورودی برابر است را برمی‌گرداند. $O(n)$
push(Integer)	گره جدیدی با مقدار داده شده در ورودی ایجاد کرده و در مکان مناسب از درخت درج می‌کند. $O(1)$ تا $O(n)$
push(Integer,Integer)	همه مقادیر ورودی را تک تک در مکان مناسب از درخت درج می‌کند.
pop()	آخرین گره از درخت را حذف می‌کند. در صورت خالی بودن درخت خطایی از نوع Underflow پرتاب می‌کند. $O(1)$ تا $O(n)$
printBFS()	پیمایش اول سطح درخت را چاپ می‌کند. $O(n)$
printDFS()	پیمایش اول عمق درخت را چاپ می‌کند. $O(n)$

بخش تعاریف ثابت

در این بخش به توضیح کلاس‌ها و تعاریفی که به صورت ثابت در اختیار شما قرار داده شده و باید بدون تغییر از آن‌ها استفاده کنید، پرداخته شده است.

فایل tester.cpp برای آزمون پیاده‌سازی شما نوشته شده است. این کلاس، دستوراتی را به شکل خلاصه شده از ورودی دریافت کرده و توابع متناظر از درخت پیاده‌سازی شده توسط شما را فراخوانی می‌کند. صحت پیاده‌سازی شما براساس مقادیری که این فایل در خروجی استاندارد چاپ می‌کند، بررسی خواهد شد. شما نیز می‌توانید با استفاده از همین فایل به تست پیاده‌سازی خود بپردازید.

دستور	توضیحات
new x	یک درخت جدید با حداکثر درجه x ایجاد می‌کند
copy	یک درخت جدید ایجاد کرده و درخت قبلی را به سازنده آن ارسال می‌کند
end	آخرین درخت ایجاد شده را حذف می‌کند
exit	تست را خاتمه می‌دهد
size	اندازه آخرین درخت ایجاد شده را چاپ می‌کند
printBFS	پیمایش اول سطح آخرین درخت ایجاد شده را چاپ می‌کند
printDFS	پیمایش اول عمق آخرین درخت ایجاد شده را چاپ می‌کند
empty	خالی بودن/نبودن آخرین درخت ایجاد شده را بررسی می‌کند
push x	تابع push را با پارامتر ورودی x برای آخرین درخت ایجاد شده فراخوانی می‌کند
pushB n x1 x2 …	تابع push را با دو پارامتر آرایه [x1,x2,…] و طول آرایه n برای آخرین درخت ایجاد شده فراخوانی می‌کند
pop	تابع pop را برای آخرین درخت ایجاد شده فراخوانی می‌کند
find x	تابع find را با پارامتر ورودی x برای آخرین درخت ایجاد شده فراخوانی می‌کند
count x	تابع count را با پارامتر ورودی x برای آخرین درخت ایجاد شده فراخوانی می‌کند

یک نمونه ورودی

```
new 2
  push 1
  push 2
  push 3
  pushB 3 4 4 6
  size
  height
  empty
  printBFS
  printDFS
  find 4
  find 7
end
```

خروجی نمونه بالا

```
6
2
false
START->1->2->3->4->4->6->END
START->1->2->4->4->3->6->END
found
not-found
```

نکات:

- نگهداری کل درخت با استفاده از آرایه و سایر ساختمان داده‌های مشابه مجاز نیست. گره‌ها باید توسط یک ساختمان داده مانند آنچه در کلاس پیاده‌سازی شده، مورد استفاده قرار گیرند.