بسمه تعالی نام درس: ساختمان داده استاد درس: دکتر حسین رحمانی پروژه شماره ۲	زمان تحویل: ۹۶/۹/۲۵ زبان پیاده سازی: C++ و Java عنوان پروژه: DNA List	 دانشکده مهندسی کامپیوتر

مدرسین حل تمرین:

- مهدیه اثنی‌عشری
- امیرمهدی میرفخار
- زهرا صادقی‌عدل
- سیداحسان سیدعلی‌اکبر
- بنفشه کریمیان
- وحید محسنی

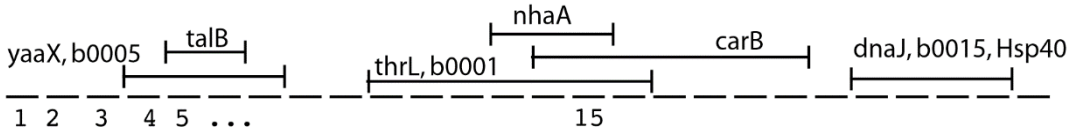
- برای فاز اول پروژه تا تاریخ ۹۶/۹/۲۰ و برای فاز دوم تا تاریخ ۹۶/۹/۲۵ فرصت خواهید داشت.
- تاریخ‌های تعیین شده قابل تمدید نمی‌باشند.
- پروژه‌ی خود را به ایمیل‌های زیر ارسال نمایید:
  - e.aliakbar97@gmail.com
  - v_mohsseni@comp.iust.ac.ir
- در subject ایمیل کلمه‌ی **Project_2a** و در متن آن نام و نام‌خانوادگی و شماره دانشجویی خود را بنویسید.
- اگر به صورت گروهی پروژه را انجام می‌دهید فقط یکی از اعضای گروه پروژه را ارسال کند.
- برای انجام این پروژه گروه‌های حداکثر ۲ نفری تشکیل دهید.
- در صورت مشاهده‌ی تقلب برای طرفین نمره -**صفر**- در نظر گرفته خواهد شد.
  برای این پروژه مجاز نیستید از کتابخانه‌های DataStructure های آماده استفاده کنید.
- برنامه شما نباید به هیچ عنوان دارای نشت حافظه باشد. برای این پروژه بجز کتابخانه iostream در c++ اجازه استفاده از هیچ کتابخانه دیگری را ندارید.
- تمامی کدهای برنامه بایستی توسط شخص شما زده شده باشد.

ژنوم انسان از مولکول‌های DNA تشکیل شده است که می‌توان آن را به صورت دنباله‌ای خطی از حروف A، C، G و T درنظر گرفت. بعضی قسمت‌های DNA برای ژن‌ها عمل رمزگزاری را انجام می‌دهند. از آن‌جاییکه ژنوم انسان بسیار بلند است، برای جستجو کردن و مطالعه درباره‌ی آن‌ها نیازمند به کامپیوتر هستیم. این پروژه وسیله‌ای برای ذخیره‌سازی و یافتن مکان ژن‌ها درطول ژنوم ارائه می‌کند. شما برای انجام این پروژه هیچ نیازی به دانستن علم زیست‌شناسی ندارید!

در فاز اول شما یک ساختمان‌داده برای نگاشت نام ژن‌ها به مکان آن‌ها ایجاد می‌کنید. در فاز دوم شما با استفاده از درخت فاصله (interval tree) ارتباط ژن‌ها با یکدیگر را تحلیل می‌کنید.

نحوه ارائه ژن‌ها

ما ژنوم را به‌صورت خطی از اعداد integer و ژن‌ها را به صورت بازه‌ی [x1, x2] در نظر می‌گیریم به نحوی که :



هر ژن حداقل یک نام دارد که به وسیله‌ی آن، می‌توانیم دسترسی پیدا کنیم.

برای نمایش هر فاصله، شما باید یک ساختار برای ژن در نظر بگیرید که حداقل فیلدهای نشان داده شده در ادامه را داشته باشد:

```
struct Gene {
   int x1, x2;          // gene from x1 to x2 inclusive
   list<string> names;  // a list of strings
};
```

"names" لیستی است که تمام نام‌های این ژن را نگه می‌دارد. متغیرهای x1 و x2 ابتدا و انتهای فاصله‌ای را که ژن پوشش می‌دهد، نگه‌داری می‌کنند. برای این پروژه شما مجاز به استفاده از کتابخانه "string" و یا هیچکدام از "STL container classes" نیستید به جز std::list و بایستی خودتان تمامی موارد لازم را پیاده سازی کنید.

فاز اول: نگاشت نام‌ها به ژن‌ها

برنامه‌ی شما باید دستورات زیر را که روی حالت استاندارد ورودی (ترمینال) ارائه می‌شود، اجرا (handle) کند. (دستورات به کاربر اجازه می‌دهند که چند نام را به یک ژن نگاشت دهد)

شما بایستی یک دیکشنری ارائه کنید که در آن هر key نام یک ژن بوده و value آن نیز خود ساختار ژن است. در این دیکشنری کلیدهای زیادی ممکن است به شی یکسانی اشاره کنند.

شما باید دیکشنری را با استفاده از "splay tree" ایجاد کنید. دستوراتی که شما باید از آن پیروی کنید به شرح زیر است:

1- AddGene name x1 x2

یک struct ژن جدید در مختصات x1 و x2 با نام داده شده تحت عنوان "name" ایجاد می‌کند و یک نگاشت بین نام و struct ژن در دیکشنری وارد می‌کند. اگر درحال حاضر یک ژن با همان نام وجود داشته باشد، برنامه باید پیغام خطا را چاپ کند.

2- AddGeneAlias name1 name2

یک نگاشت جدید بین name2 و ژنی که با name1 نگاشت شده بود ایجاد می‌کند. هم‌چنین name2 را به عنوان یکی از نام‌ها در "name list" ژن وارد می‌کند. اگر "name2" درحال حاضر به هرکدام از ژن‌ها اختصاص داشت و یا "name1" به هیچ ژنی تعلق نداشت، پیغام خطا باید نمایش داده شود و هیچ تغییری نباید صورت گیرد.

3- RemoveAlias name

"name" را از دیکشنری و "names list" مناسب پاک می‌کند. اگر "names list" خالی شد، ژن همراه با آن نیز باید پاک شود (آزاد شود). بنابراین ژن‌ها با پاک شدن تمامی نام‌های اختصاص یافته به آن‌ها پاک می‌شوند. اگر نام داده شده وجود نداشت برنامه باید پیغام خطا را چاپ کند.

4- PrintGeneInfo name

باید مختصات و نام‌های ژن داده شده را با فرمت زیر چاپ کند:

```
x1 x2 alias1 alias2 alias3 ...
```

5- Quit

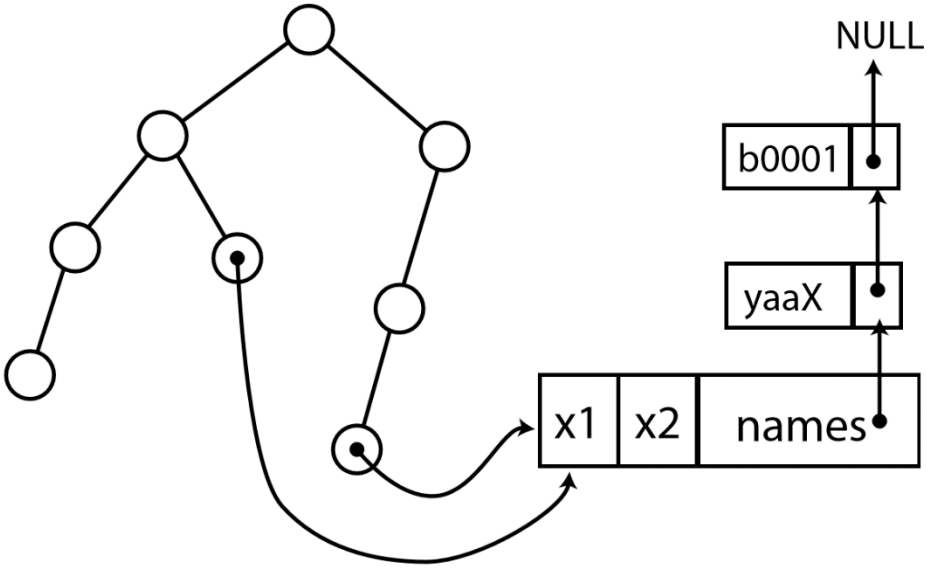
باید برنامه را تمام کند.

- پیغام خطا:

هنگام بروز خطا، شما باید پیغامی با محتوای زیر در خروجی چاپ کنید:

"Error: " + توضیحی کوتاه درباره خطا

خطا باید در یک خط چاپ شود.



قسمت دوم: بررسی "Gene regularity graph"

بعضی از ژن‌ها سایر ژن‌ها را خاموش و یا روشن می‌کنند. اگر ژن "A" ژن "B" را خاموش و یا روشن کند، میگوییم "A"، "B" را تنظیم می‌کند. این روابط دوبه‌دو می‌توان یک گراف جهت‌دار را با رئوس ژن‌ها و یال‌ها به صورت زیر مشخص کنند:

ژن A با ژن B ارتباط دارد اگر A، B را تنظیم کند.

شما باید یک ساختمان داده‌ای ارائه دهید که گراف اسپارس و جهت‌دار فوق را با توجه به دستورات زیر ایجاد کند.

1- Regulates name1 name2

یک یال جهت‌دار از ژن با نام اول یه ژن با نام دوم ایجاد می‌کند. اگر هرکدام از نام‌های 1 یا 2 یک نام معتبر برای ژن‌ها نبود، برنامه پیغام خطا می‌دهد.اگر یال جهت دار گفته شده موجود بود، برنامه کاری انجام نمی‌دهد.

2- DonateRegulate name1 name2

یال جهت دار موجود از ژن با نام 1 به ژن با نام 2 را حذف می‌کند. اگر هرکدام از نام‌ها معتبر نبود، برنامه پیغام خطا چاپ می‌کند و درصورتی که یالی وجود نداشت، برنامه هیچ کاری نمی‌کند. گراف برنامه شما به هیچ وجه نباید راس ایزوله داشته باشد. درصورتیکه هر یک از ژن‌ها تنظیم‌کننده ژن دیگری نبود و یا توسط ژن‌های دیگر تنظیم نمی‌شد نباید در گراف نمایش داده شود.

3- PrintGraph

گراف را با فرمت زیر چاپ می‌کند:

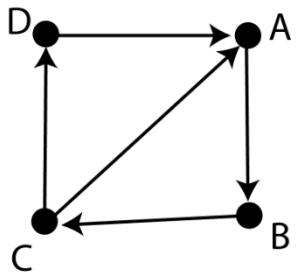
```
Disgraph G {
    name1 -> name2
}
```

نام اول و دوم نام ژن‌هایی هستند که با هم ارتباط دارند. توجه شود که اگر ژنی چند نام داشت می‌توان به صورت تصادفی یکی از نام‌های آن را نمایش داد اما درصورتیکه هرنام نمایش داده شد در دفعات بعدی نیز باید همان نام نمایش داده شود.

4- PrintFeedBackLoops name k

با استفاده از bfs تمامی دورهای ساده جهت‌دار را که به ژن متناظر با "name" به گونه ای تعلق دارد که :

اندازه دور کوچکتر از k باشد.




```
PrintFeedBackLoops A 3
A B C
PrintFeedBackLoops A 4
A B C ; A B C D
```