بسم اللّه الرحمن الرحیم

تمرین های سری سوم درس برنامه نویسی پیشرفته کامپیوتر

امیر جهانشاهی

۲۹ اسفند ۱۳۹۷

تمرین کارگاه Frontend

سلام بچه ها عیدتون حسابی مبارک باشه بچه ها. تیم تدریس یاریِ سرشار از تدریس یارتون سال خوشی رو براتون آرزومنده و دعا میکنه همتون از دولوپرهای خفن گوگل و اپل و مایکروسافت بشید البته مهم تر از اون امیدواریم کانتریبیوشن اورویوی گیت هابتون سبز بهاری باشه.

برای عید تمرین های باحالی براتون درنظر گرفتیم که اولش با یه بازی شروع میکنیم. از قدیمی ترین بازی های گروهی ایرانی همون نقطه بازی خودمونه که همتون بلدید قطعا ولی برای اونایی که بلد نیستن یه توضیح کوتاه میدم.تو صفحه چندتا نقطه میذاریم مثلا هفت در پنج و باید به نوبت اینارو به هم وصل کنیم هرکی که خونه ی ۴ تایی (یعنی یک در یک) رو تکمیل کنه اون خونه رو برنده میشه و یه امتیاز میگیره، جایزه اش هم اینه که دوباره نوبت خودشه، درضمن برنده ی هر خونه باید خونه رو رنگ خودکار خودش کنه(آبی یا قرمز مثلا). حالا میخوایم این بازی رو با وب (فرانت اند) و علی الخصوص با جاوااسکریپت پیاده سازی کنیم.استفاده ی هیچ کتابخونه ای هم به جز جی کوئری مجاز نیست(مثلا متریالایز هم مجاز نیست). اول بازی یک صفحه میاد که اسم دو نفر رو میگیره و ابعاد صفحه رو هم میگیره (مثلا دو در سه) و صفحه عوض میشه وصفحه ی بازی ساخته میشه.هر کسی باید بین نقطه ها کلیک کنه و خطش کشیده بشه وبصورت خودکار نوبت نفر بعدی بشه. درنهایت هم وقتی پر شد صفحه بازی تموم بشه و برنده اعلام بشه.

نکته ی اول: زیبایی صفحه هم تو امتیازتون تاثیر داره.

نکته ی دوم: مینیمالیستی (کمترین استفاده از رنگ، عکس و هر المانی) جزو سبک های بسیار جذاب و ترند زیبایی هست.

نکته ی سوم: بازی باید ریسپانسیو باشه که بعدا تدریس یارا بتونن تو گوشی باهاش بازی کنن عیدشونو بگذرونن.

راهنمایی اول: میتونید همه چیز رو یه المان(دیو، باکس) در نظر بگیرید و مثلا یه دیو ۳پیکسل در سه پیکس مشکی بشه نقطه اتون و یه دیو ۳ در ۱۰ بشه خطتون و دیو ۱۰ در ۱۰ بشه خونه اتون.

راهنمایی دوم: برای ریسپانسیو کردن دوتا چیز توصیه میشه اما اجبار نیست:

۱. برای تغییر ارتفاع با سایز صفحه به جای واحد پیکسل از واحد vh استفاده کنید.

۲. برای تغییر طول هر خانه میتونید با جاوااسکریپت راحت طول صفحه رو بگیرید و تقسیم بر تعداد کنید (یعنی طول رو تو سی اس اس ندید با جاوااسکریپت تولیدش کنید).

بخش امتیازی: اگر اول بازی تعداد افراد شرکت کننده رو هم بپرسه و به اون تعداد بازی رو بسازه که عالیه !

تمرین C++

۱. در این سوال قصد داریم MaxHeap را پیاده‌سازی کنیم. ساختمان داده MaxHeap ، یک درخت باینری‌ست که در آن کلید والد بزرگتر یا مساوی با مقدارکلید فرزندانش است. برای آشنایی بیشتر به این لینک مراجعه نمایید. کلاس MaxHeap باید شامل توابع عمومی زیر باشد:

add(key) که گره با مقدار key را به درخت در جایگاه مناسب، اضافه می‌نماید.

Delete() که در هربار فراخوانی، گره با مقدار max را از درخت حذف کرده و درخت را مرتب می‌نماید تا دوباره Maxheap ساخته شود.

Max() که کلید گره با بزرگترین مقدار را باز می‌گرداند.

getHeight() که ارتفاع درخت را بازگردانی می‌کند.

Parent(i) که مقدار کلید والد گره i م را باز می‌گرداند.

LeftChild(i) ، RightChild(i) که به ترتیب مقادیر فرزند سمت راست و فرزند سمت چپ را بازمی‌گرداند. یکی از کاربردهای این ساختمان داده، مرتب‌سازی آرایه با پیچیدگی زمانی O(nlogn) است. تابع Heapsort را به گونه‌ای پیاده‌سازی کنید که مقادیر Heap را به صورت نزولی چاپ کند. توجه کنید که پس از اعمال heapsort ، ساختار Heap خراب می‌شود.

printArray() که آرایه‌ی maxheap را چاپ می‌نماید.

Operator [i] که مقدار کلید گره i م را باز می‌گرداند.

Operator +/- که مقدار maxheap[i] را با یک مقدار جمع/تفریق می‌کند و اگر ساختار maxheap بهم ریخت، آن را مرتب می‌نماید.

Operator« که maxheap را به فرم درخت چاپ نماید. (مانند نمونه خروجی ۱)

راهنمایی: برای پیاده‌سازی ساختار Maxheap از یک آرایه، نیاز به پیاده‌سازی توابع max_heapify() و build_max_heap() و فراخوانی آنها در Constructor دارید.

۲. در این سوال می خواهیم کلاسی مشابه با وکتور را پیاده سازی کنیم. این کلاس شامل دو تابع

```
64,
53, 59,
42, 34, 56, 24,
18, 27, 25, 22, 40, 20, 10, 5,
8, 9,

Press any key to continue . . .
```

شکل ۱: نمونه خروجی سوال یک

```
int main()
{
    Maxheap h1{};
    h1.add(25);
    h1.add(32);
    h1.add(17);
    h1.add(23);
    h1.add(101);

    std::cout << h1 << std::endl;
    /*
    101,
    32, 17,
    23, 25
    */
    std:cout << h1.parent(2) << std::endl; // 101
    std::cout << h1.LeftChild(0) << std::endl;  // 32
    std::cout << h1.RightChild(0) << std::endl;  // 17


    int arr[7] {23, 1, 7, 52, 11, 10, 75};
    Maxheap h2{arr, 7};

    std::cout << h2 << std::endl;
    std:cout << h2.getHeight() << std::endl; // 2


    Maxheap h3{h2};

    h2.Heapsort();
    h2.printArray(); // 75, 52, 23, 11, 10, 7, 1


    std::cout << h3.max(); //75

    h3[0] = h3[0] - 25;

    std::cout << h3.max(); //52
    h3.delete();
    std::cout << h3.max(); //50

```

شکل ۲: تابع main سوال یک

push_back و pop_back برای اضافه کردن یک مقدار به انتهای وکتور و یا پاک کردن آخرین خانه وکتور می‌باشد. این کلاس را به این صورت تعریف کنید که با هر بار push کردن size آرایه دینامیکی که در آن مقادیر ذخیره می شوند یکی اضافه شود اما capacity آن به این صورت باشد که با اضافه کردن سومین داده ۴ شود و هم چنین با اضافه کردن ۵ امین داده ۸ شود و روند به همین ترتیب ادامه یابد. با هر بار pop کردن نیز باید از سایز کاسته شود و وقتی مثلا سایز از ۸ به ۴ رسید آن گاه capacity نیز کاهش یابد (این مقادیر به گونه‌ای تعریف شوند که بتوان به مقدار آن ها در main دسترسی داشت).
سپس متود های جمع و ضرب داخلی را نیز برای این وکتور ها تعریف کنید. این وکتورها را نیز از نظر اندازه آن ها با اپراتور های > = < مقایسه کنید. copy constructor و move constructor را نیز برای این کلاس تعریف کرده و فرق بین این constructor ها را در گزارش شرح دهید. در انتها نیز تابع show را برای نمایش هر کدام از وکتور ها تعریف کنید. راهنمایی: از یک متغیر int * برای ذخیره داده‌ها در کلاس استفاده کنید و دقت کنید که با هر بار push یا pop کردن طول آرایه دینامیکی عوض می‌شود و نیاز است که آرایه مجددا ساخته شود و آرایه قبلی حتما پاک شود.

۳. با نزدیک شدن به سال جدید مردم شهر محبت که ماه ها بود درگیر شرایط اقتصادی بد بودند، دچار یک حمله شدید روحی شدند. طولی نکشید در پی یک اتفاق نادر پس از چند روز تمامی مردم شهر دچار بیماری فراموشی شدند و سردرگرم و جیغ زنان در خیابان ها می دویدند. از آن جا که شروین یک developer حرفه ای است و هیچگاه از لپتاپ و اتاقش جدا نمی شود همیشه ی خدا از همه ی اتفاق های شهرش بی خبر است تا این که امروز صدای جیغ مردم به حدی بالا گرفت که به خود زحمت داد و پنجره ی اتاقش را باز کرد! " چه اتفاق افتضاحی!"

پس از ماه ها حبس خود در اتاق به سرعت به سمت آشپزخانه رفت و با مادرش در حال گریه روبه رو شد. چندی نگذشت که مطلع شد نه تنها پدر و مادرش بلکه همه ی مردم شهر، تمام خاطرات و افراد مهم زندگیشان را فراموش کردند. شروین که از این بیماری جان سالم به در برده بود تصمیم گرفت به عنوان یک هکر کلاه سفید تمام اطلاعات شخصی خانواده هارا جمع آوری کند و همه ی خانواده ها را به حالت قبل بازگرداند تا دوباره محبت بینشان جوانه بزند. :)

در پی این تصمیم شروین می خواهد با کمک شما دو کلاس برای مردم و خودش با نام های Human و Oracle طراحی کند تا بتواند به این ترتیب تمامی اطلاعات موجود را دسته بندی کند.

```
class human {
..
human(std::string firstName, std::string lastName, int hairColor, int eyeColor, int
age, bool gender, int numberOfChildren);
std::string getFirstName();
std::string getLastName();
bool getGender();
int getHairColor();
int getEyeColor();
int getAge();
bool operator >(human *, human *);
bool operator ==(human *, human *);
human* operator+(human *, human *);
void operator++();
bool isChildOf(human *);
bool isFatherOf(human *);
```

```
bool isMotherOf(human *);
void printChildren();
..
};
```

کانستراکتور کلاس تمامی مشخصات هر فرد را مشخص می کند. توجه کنید این مشخصات باید به صورت متغیر های private باشد.

هر شخص یک human* father ، human* mother و human* spouse دارد که به ترتیب مشخص کننده ی پدر و مادر و همسر این فرد هستند. همچنین هر فرد یک لیستی از human* ها دارند که نشان دهنده ی فرزندان آن هاست. اسم و فامیل هر شخص به صورت string و با حروف بزرگ می باشند. hairColor و eyeColor هرکدام اعدادی بین ۰ تا ۱۰ می باشند که هر کدام نشان دهنده ی رنگ خاصی هستند(خودتان را درگیر اینکه هر کدام چه رنگی است نکنید.) وقتی gender ، false است نشان دهنده ی جنسیت مرد و در صورت true بودن نشان دهنده ی جنسیت زن است. numberOfChildren نیز تعداد فرزندان را مشخص می کند. توابع get نوشته شده مقادیر ویژگی ها ی هر فرد را برمیگرداند.

bool operator >(human *); این تابع برای مقایسه کردن سن دو انسان است.

bool operator ==(human *); در صورتی که تمام attribute های دو انسان یکی باشند true برمی گرداند.

human* operator+(human *); در این اپراتور اگر دو انسان همسر یک دیگر باشند یک human جدید با سن صفر تولید می کند و هریک از ویژگی های دیگرش را به صورت رندوم از زن و مرد دریافت می کند (البته فامیلی فرد جدید باید شبیه با فامیلی مرد باشد.) در نهایت این انسان در لیست فرزندان زن و مرد قرار گرفته و این زن و مرد به صورت متقابل پدر و مادر این فرد می شوند.(دقت داشته باشید که همه ی کارهای ذکر شده باید در این operator انجام شود)

void operator++(); سن فرد مورد نظر را یکی اضافه می کند.

bool isChildOf(human *); مشخص می کند که آیا this فرزند ورودی است یا خیر.

bool isFatherOf(human *); مشخص می کند که آیا this پدر ورودی است یا خیر.

bool isMotherOf(human *); مشخص می کند که آیا this مادر ورودی است یا خیر.

void printChildren(); اسم تمامی فرزندان این شخص را چاپ می کند. (دقت کنید اسم افراد به صورت نزولی براساس سنشان چاپ شود.) دقت کنید به دلیل وجود human** که آرایه از human* است که هر کدام به یک فرد اشاره دارد. بنابراین نیاز به copy constructor و operator = است تا double free رخ ندهد. هم چنین این لیست باید در destructor نیز باید پاک شود. تعداد فرزندان در

متغیر numberOfChildren نگهداری می شود.

```
class Oracle {
..
Oracle(std::string Name);
Bool marry(human*, human*);
Bool isFamily(human*, human*);
Void setChild(human*, human*, human*);
Human** getFamily(human*);
int getPopulationOfFamily(human*);
..
};
```

Oracle(std::string Name); این تابع کانستراکتور این کلاس است که اسم oracle را دریافت میکند.

Bool marry(human*, human*);

در صورت اینکه این دو قبلا با هم ازدواج نکرده باشند و مجرد باشند (سن آن ها بزرگتر مساوی ۱۸باشد) می توانند ازدواج کنند و باید هرکدام در متغیر Spouse دیگری قرار بگیرند . اگر ازدواج قابل انجام باشد خروجی true می شود در غیر این صورت جواب false است.

Bool isFamily(human*, human*); اگر این دو شخص به هر نحوی با یکدیگر فامیل باشند(برادرزن پسر عمو هم حتی فامیل حساب می شود!) باید true برگرداند.

Human** getFamily(human*); این تابع آرایه ای از تمام افرادی که به هر نحوی با این شخص فامیل باشند، بر می گرداند. (مراقب باشید یک فرد یکسان دوبار تکرار نشود!)

Void setChild(human*, human*, human*); این تابع ورودی اول را فرزند دو ورودی بعدی قرار می دهد و بالعکس ورودی بعدی را پدر و مادر ورودی اول قرار می دهد. (ورودی دوم مادر و سوم پدر است)

int getPopulationOfFamily(human*); این تابع تعداد افراد یک خانواده را برمی گرداند. (برای مثال نوه های پدر بزرگ پسر عموی شما هم جزو خانواده به حساب می آیند!!!)

دقت کنید تنها اجازه دارید از این توابع به همین صورت که نامگذاری شده اند استفاده کنید و هر گونه نام گذاری متفاوت مردود است. (البته تعریف متغیر های کمکی بلامانع است) تمام ویژگی های بالا باید در تابع main که در فایل تمرینات قرار داده شده است، نشان داده شود.

به شروین کمک کنید تا شهر خود را دوباره بسازد.

تمرین کارگاه Database

تو تمرین قبلی فرانت اند سعی کردید یه شمای استاتیک از تلگرام بسازید؛ پاول دورف دید هر چقدر رو جاهای مختلف صفحه کلیک می کنه اتفاقی نمی افته و حسابی از دست شما عصبانی هستش و فک کرد ما داریم دست می ندازیمش! برای این که دوباره بتونید دلش رو به دست بیارید حالا می خوایم با کمک هم قسمت بک اندش هم کم کم بیاریم بالا.
چون هنوز بلد نیستید که شمای دیتابیس را خودتان طراحی کنید قرار هست ما هم کمکتان کنیم. شمای زیر (مشخصات table ها) را در نظر بگیرید. اینجا قرار هست بعضی از ویژگی های تلگرام را پیاده سازی کنیم.

User(id, telegram_id, phone, email, password, first_name, last_name, verification_code, profile_picture_url, created_at, updated_at)

منظور از telegram_id یک id از نوع string هستش که با @ می توانید آن را منشن بکنید و منظور از id یک عدد یکتا هست. فرض شده هر کاربر فقط می تواند یک عکس پروفایل داشته باشد. همچنین password ها را به صورت hash نگهداری کنید (این کار را در موقع insert کردن می توانید انجام دهید).

BlockUser(blocker_user_id, blocked_user_id, ecreated_at)

Message(id, sender_id, receiver_id, message_type, message_text, created_at, updated_at)

ستون message_type می تواند مقادیر صوتی، ویدیو، عکس و نوشتاری را داشته باشد. در صورت غیر نوشتاری بودن پیام، message_text در حکم caption خواهد بود.

Channel(id, telegram_id, title, info, creator_id, created_at, updated_at)

ستون creator_id درواقع به یکی از اعضای User اشاره می کند و فرض شده هر کانال فقط می تواند یک سازنده (همان نویسنده پیام ها) را داشته باشد. همچنین منظور از updated_at آن است که مشخصات کانال آخرین بار چه زمانی آپدیت شده است و ارتباطی با زمان پست ها ندارد.

Group(id, creator_id, title, join_url, created_at, updated_at)
GroupMessage(id, group_id, sender_id, message_type, message_text, created_at, updated_at)
ChannelMessage(id, channel_id, message_type, message_text, created_at, updated_at)
MessageAttachment(message_id, attachment_url, attachment_thumb_url)

اطلاعات پیوست ها در اینجا آورده می شود. برای پیام هایی که حاوی اطلاعات غیر متنی هستند فایل مربوطه در جایی از سرور آپلود خواهد شد و آدرس آن در اینجا قرار می گیرد.

GroupMessageAttachment(message_id, attachment_url, attachment_thumb_url)
ChannelMessageAttachment(message_id, attachment_url, attachment_thumb_url)
ChannelMembership(user_id, channel_id, created_at)
GroupMembership(user_id, group_id, created_at)

در بالا سه نوع پیام داریم. اولی پیام های Message که در واقع بین دقیقا دو کاربر رد و بدل شده است (اصطلاحا PV). دومی پیام های Group ها و سومی پیام های Channel ها.

۱. جداول بالا را در PostgreSQL بسازید. برای هر یک از جدول ها در گزارش خود مشخص کنید که primary key و foreign key کدام ستون (ها) هستند و foreign key هر جدول به کدام جدول اشاره دارد. کد سازنده ی جدول های خود را در فایلی با نام create.sql در کنار فایل های خود قرار دهید.

۲. برای هر یک از کوئری های زیر، شرایطی خاص ایجاد کنید (از طریق insert کردن) تا بتوان نحوه ی عملکرد صحیح کوئری را ارزیابی کرد؛ کد insert هایی که برای هر کوئری انجام می دهید را در فایلی با نام insert.sql در کنار فایل های خود قرار دهید.

۳. کوئری های زیر را به زبان SQL بنویسید و سپس کد آن ها را در فایلی با نام query.sql قرار بدهید. در این سری تمرین سعی شده تا کوئری های جدی تر مطرح نشوند تا زمانی که در جلسه ی بعدی بیشتر در رابطه با nested query آشنا بشوید و دوباره به این تمرین باز می گردیم.

۱ـ مشخصات کاربر به شماره تلفن ۰۹۱۲۰۰۰۰۰۰۱ را برگردانید؛ واقعا کنجکاو هستیم صاحب این شماره جذاب را بشناسیم!

۲ـ ایمیل کاربر با شماره تلفن ۰۹۱۲۰۰۰۰۰۰۱ را به apstudent2019@gmail.com تغییر دهید؛ از قضا از دانشجویان خودمان هستند.

۳ـ نام کانال هایی که کاربر با شماره تلفن ۰۹۱۲۰۰۰۰۰۰۱ عضو می باشد را برگردانید؛ فضولی مان گل کرده است و می خواهیم سایر علایق این دوست عزیز را بدانیم.

۴ـ تعداد فالور های (اعضا) کانال با آی دی تلگرامی aut_ap_2019 را برگردانید.

۵ـ ایمیل کاربرهایی که شماره تلفن آن ها با پیش شماره ۰۹۳۵ شروع می شود را برگردانید چون ایرانسل قرار است بین آن ها قرعه کشی انجام دهد.

۶ـ شماره تلفن کاربرهایی را برگردانید که کاربر با شماره تلفن ۰۹۱۲۰۰۰۰۰۰۱ آن ها را در یک ماه اخیر بلاک کرده است تا از آن ها علت را جویا شویم که چرا کاربر عزیز ما را دلخور کرده اند.

۷ـ ایمیل صاحبان کانال هایی که اعضای آن بیشتر از ۳ عضو دارد و کاربر با ایمیل apstudent2019@gmail.com نیز در آن ها عضو است را برگردانید.

۸ـ ۱۰ پیام آخر مکالمه ی دونفره ی بین دو کاربر با آی دی های تلگرامی sHDiV4RHs و amir.jahanshahi را برگردانید تا بفهمیم در پشت صحنه چه می گذرد؛ سپس در کوئری دیگری تمامی پیام های بین این دو شخص را پاک کنید.

۹ـ شماره تلفن کاربران صاحب کانال هایی را برگردانید که در یک ماه اخیر هیچ پستی در کانال آن ها گذاشته نشده است (از تاریخ آخرین پست بیش از یک ماه گذشته باشد)؛ شاید نیاز به جا به جایی اطلاعات این کانال ها به سروری دیگر باشد.

از خروجی ترمینال دیتابیس خود برای هر یک از کوئری ها اسکرین شات بگیرید و در گزارش قرار دهید. همچنین اطلاعات فعلی دیتابیس خود را export کنید و در کنار فایل های خود قرار بگذارید.

۴) به نظر شما چه تغییری باید در طراحی داد تا بتوان بیش از یک ادمین برای هر کانال یا گروه داشت که توانایی پست گذاشتن هم داشته باشند (برای کانال).

حتی الامکان ستونی به جداول اضافه یا از آن ها کم نکنید تا کار تصحیح دشوار نشود و در صورتی که به دلیل ناقص بودن مدل از نظر کارایی تصمیم به چنین کاری گرفتید لطفا ابتدا در گروه مطرح کنید. شاید چند کوئری آخر کمی برایتان سخت تر به نظر بیاید. سعی کنید اول روی کاغذ به جواب مورد نظر برسید؛ همچنین اگر مشکلی داشتید لطفا بیان کنید.

جهت تحویل تمارین، هر تمرین را داخل یک فولدر بریزید که با شماره تمرین نام گذاری شده است. گزارش کار را به صورت PDF در فولدر اصلی تمرین ها قرار دهید. در نتیجه در فولدر Q1, Q2, ... اصلی فقط یک فایل گزارش موجود می باشد و تعدادی فولدر که با شماره تمرین ها نام گذاری شده است. اسم فلدر اصلی را به صورت زیر نام گذاری و سپس فشرده سازی و در قالب یک فایل ارسال کنید. توجه نمایید که از قالب فشرده سازی rar استفاده نکنید.

پاسخ تمرین‌های خود را در یکی از سرویس های github و یا gitlab در یک repository به نام AP-HW2 به صورت Private بارگذاری نمایید. برای این کار باید در قسمت New repository در زمان ساختن repository جدید حالت Private را انتخاب نمایید.

در ادامه تمرینات انجام شده را با فولدر بندی مناسب (سوال ۱ داخل فولدری به همین نام و ...) داخل این پروژه آپلود نمایید. در بخش گزارش فرآیند بارگذاری را شرح دهید و لینک تمرین را داخل گزارش ذکر نمایید.

دقت کنید که با توجه به موارد گفته شده، فایل gitignore. را به نحویی طراحی کنید که تنها فایل های اصلی و make file درون git قرار داده شوند.

توجه: به منظور دسترسی به تمرین برای تصحیح، پس از پایان زمان تحویل تمارین پروژه را به حالت Public تغییر دهید.

AP-HW3-شماره دانشجویی.zip

مهلت تحویل: تا ساعت ۲۳، ۲۰ فروردین ماه ۱۳۹۸