

## گزارش تمرین سری سوم درس برنامه نویسی پیشرفته

### محمد صالح گشانی – 9423092

در کد های فرستاده شد برای فهم بهتر کد سعی شد تا جای ممکن برای دستورات کامنت گذاری شود .

#### سوال اول (C++)

در این سوال باید ساختار heap را پیاده سازی می کردیم . ساختار به این صورت است که در مقادیر آرایه را به صورت سطر سطر مرتب کرده و تعداد مقادیر هر سطر برابر بود با 2 به توان شماره ی آن سطر منهای 1 ، سپس با یک الگوریتم که چند بار تکرار می شد شروع به مرتب سازی heap کردیم . به این صورت که از سطر یکی مانده به آخر شروع کرده و نود اصلی را با هر یک از زیری آن مقایسه کردیم و در صورت نیاز جابجایی های لازم را انجام دادیم . سپس این کار را برای تمامی نود های اصلی که در همان سطر قرار داشتند انجام دادیم . در مرحله بعد همین کار را با سطر بالایی (سطر دو تا مانده به آخر) انجام دادیم و همین طور این روند را ادامه داده تا به آخر رسیدیم با این کار عمل heapify انجام شد . برای پیاده سازی این کار دو تابع build\_max\_heap و max\_heapify تعریف شد . در تابع اول تک تک مقادیر آرایه به جز مقادیر سطر آخر index شده و برای مقایسه و در صورت امکان جابجایی با مقادیر اصلی (مقدار نود بالایی) به تابع max\_heapify فرستاده می شدند . در تابع max\_heapify هر مقدار با مقادیر فرزندان (مقدار زیری چپ و مقدار زیری راست) مقایسه می شد و در صورت نیاز تعویض های لازم صورت می گرفت .

در نوشتن تابع add نکته ی قابل توجه این بود که باید capacity (حداکثر تعداد مقادیر ممکن با توجه به تعداد فعلی نود ها) نسبت به تعداد نود های جدید (که با توجه به اضافه شدن یک مقدار جدید یکی اضافه شده است) دوباره محاسبه می شد . همچنین باید مقادیر آرایه delete شده و با دوباره ساختن آرایه با اندازه ی جدید مقادیر قبلی و نود جدید در آن ریخته می شد و در پایان هم باز باید تابع build\_max\_heapify در آن اجرا می شد که با توجه به نود اضافه شده heap جدید ساخته شود . در تابع delete هم همان موانع تابع add را پیش رو داشتیم فقط یک نود از تعداد نود ها کم می شد .

در تابع getHeight با توجه به فایل main فرستاده شده اینطور به نظر آمد که خروجی مد نظر برای این تابع تعداد سطرهای درخت (در واقع عمق درخت به جز ریشه) است پس تعداد سطر ها را حساب کرده و آنرا را به عنوان خروجی بازگرداندیم (لازم به ذکر است چون تعداد نود ها در ادامه نیز نیاز می شد تابعی به نام heapHeight نوشته شد که آن تابع تعداد نود ها را به عنوان خروجی می داد).

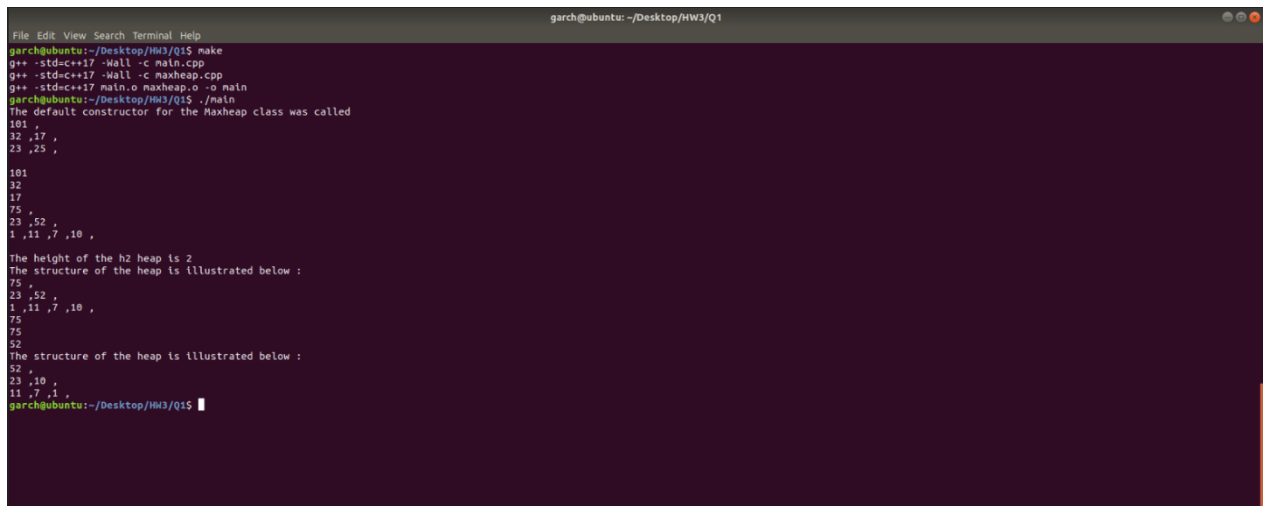
در تابع heapsort هم تنها تابع build\_max\_heapify اجرا می شد که عمل heap سازی را روی آرایه انجام می داد و در تابع printArray هم با محاسبه ی تعداد سطر ها مقادیر موجود در هر سطر چاپ می شد .

از آنجا که در تابع main در دو جا خود heap ها را cout کرده ایم لازم بود یک تابع با ورودی های cout و heap و خروجی به صورت heap نوشته شود که heap را cout کند . برای این کار یک تابع در فایل maxheap.cpp نوشتیم که این فایل عمل مورد نظر را انجام داده و در نهایت متغیر OS که متغیر از جنس ostream بود را به عنوان خروجی می داد . نکته ی قابل

توجه در این بود که با توجه به این که این تابع `heap` به عنوان ورودی می گرفت باید کلاس `heap` را قبل از این تابع `declare` می کردیم .

برای تست کد هم از همان فایل `Q1main` که همراه با تمرین فرستاده شده بود استفاده شد (خطا های `syntax` این فایل گرفته شده و آن خط کد که مربوط به تست اپراتور های `+/-` بود به دلیل حذف شدن از تمرین کامنت شد) .

یک نمونه از اجرای برنامه در شکل 1 آمده است :



```
garch@ubuntu: ~/Desktop/HW3/Q1
garch@ubuntu:~/Desktop/HW3/Q1$ make
g++ -std=c++17 -Wall -c main.cpp
g++ -std=c++17 -Wall -c maxheap.cpp
g++ -std=c++17 main.o maxheap.o -o main
garch@ubuntu:~/Desktop/HW3/Q1$ ./main
The default constructor for the Maxheap class was called
101 ,
32 , 17 ,
23 , 25 ,
101 ,
32 ,
17 ,
75 ,
23 , 52 ,
1 , 11 , 7 , 10 ,
The height of the h2 heap is 2
The structure of the heap is illustrated below :
75 ,
23 , 52 ,
1 , 11 , 7 , 10 ,
75 ,
75 ,
52 ,
The structure of the heap is illustrated below :
52 ,
23 , 10 ,
11 , 7 , 1 ,
garch@ubuntu:~/Desktop/HW3/Q1$
```

شکل 1 – نمونه اجرای تمرین 1

## سوال دوم (C++)

در این سوال باید در قدم اول هنگام نوشتن دو تابع `push_back` و `pop_back` باید دقت می کردیم که اگر `capacity` آرایه دچار تغییر می شد باید از یک متغیر اضافی برای ذخیره مقادیر فعلی آرایه استفاده کرده سپس با پاک کردن آرایه ی دینامیک و تغییر اندازه ی مقادیر قبلی را دوباره در آن ریخته و با توجه به نوع تابع (`push_back` و یا `pop_back`) مقدار بعدی را در آرایه ریخته و یا مقدار آخر آرایه را پاک کنیم .

در کد نوشته علاوه بر اپراتور های خواسته شده اپراتور `[]` برای راحتی کار با آرایه هنگام تست کد نوشته شد . علاوه بر این اپراتور `=` در دو حالت `copy` و `move` نوشته شد که در صورت نیاز هنگام تست از آن استفاده شود .

در این کلاس یک تابع کمکی به نام `incSize` نوشتیم که از این تابع در اپراتور های جمع و ضرب داخلی کمک گرفتیم . در اپراتور های جمع و ضرب در صورت هم اندازه نبودن دو آرایه باید به انتهای آرایه ی کوچکتر تعدادی صفر اضافه کرده سپس عمل مورد نظر را انجام دهیم . عمل اضافه کردن صفر و متعاقبا تغییر `size` و `capacity` آرایه توسط تابع `incSize` انجام شد .

در این کلاس `copy constructor` و `move constructor` ساخته شدند . فرق این دو در این است که در `copy` آرایه ی اولیه ای که از روی آن یک آرایه ی جدید را می سازیم آرایه ای است که وجود خارجی دارد و `lvalue` است در حالی که در `move` آرایه ی اولیه مقدار است که وجود خارجی ندارد و `rvalue` است (برای مثال در تابع `main` متغیر `v1 + v2` که `rvalue` است در آرایه ی جدید `v4` ریخته شده است) . با توجه به تفاوت مطرح شده در `move` برخلاف `copy` که آرایه ی `var` را ابتدا

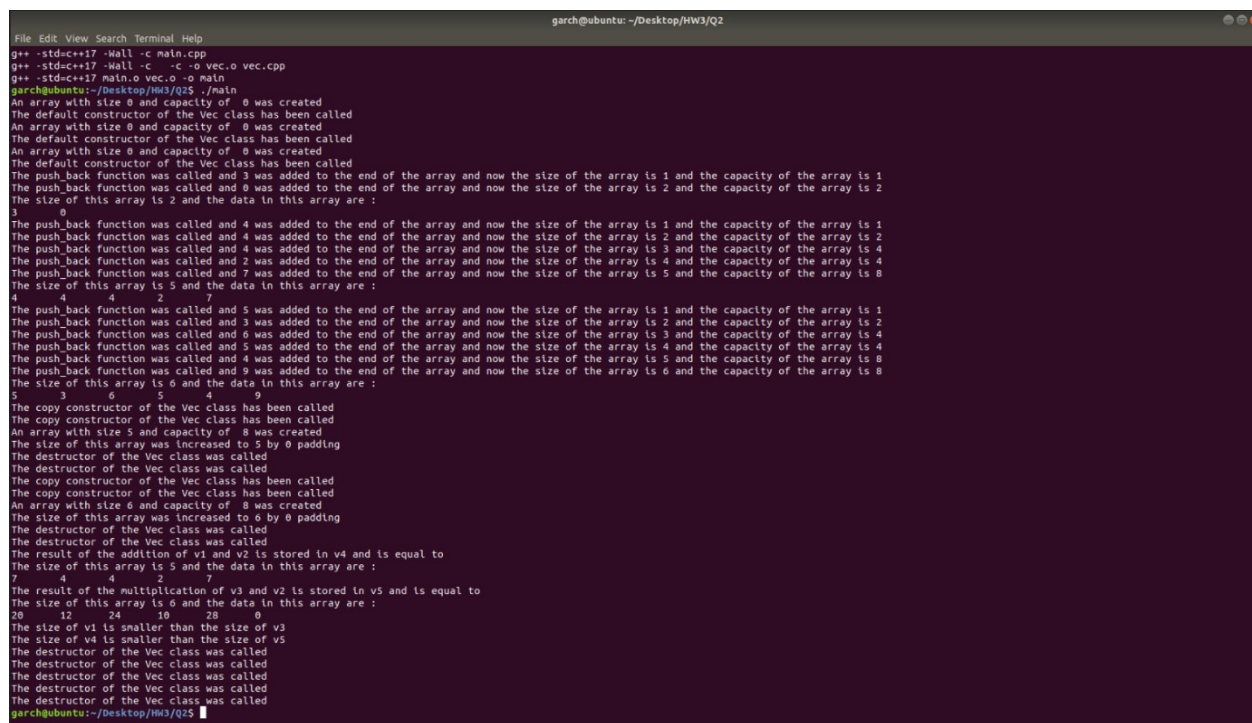
new کرده (مقداری از حافظه را رزور می کنیم) سپس آرایه مورد نظر را در حافظه ی new شده ذخیره می کنیم ، بلافاصله آرایه ی مورد نظر را در var آرایه ای که می خواهیم آن را بسازیم ریخته و با توجه به این که آرایه ی اولیه یک rvalue است مقدار آرایه ی آن را برابر با nullptr قرار می دهیم تا به چیزی اشاره نکند .

در destructor کلاس هم اگر size آرایه صفر نبود پوینتر را delete می کردیم در غیر اینصورت از آنجا که پوینتر حاوی nullptr بود به آن کاری نداشتیم .

لازم به ذکر است در تمامی توابع این کلاس اگر جایی از آرایه دینامیک استفاده شد در پایان scope مورد نظر آن ، آرایه delete شد .

همچنین در تمامی توابع متن هایی cout شد تا اجرای توابع و نتایج آن ها را نمایش دهد .

یک نمونه از اجرای برنامه در شکل 2 آمده است .



```
File Edit View Search Terminal Help
g++ -std=c++17 -Wall -c main.cpp
g++ -std=c++17 -Wall -c -c -o vec.o vec.cpp
g++ -std=c++17 main.o vec.o -o main
garch@ubuntu:~/Desktop/HW3/Q2$ ./main
An array with size 0 and capacity of 0 was created
The default constructor of the Vec class has been called
An array with size 0 and capacity of 0 was created
The default constructor of the Vec class has been called
The push_back function was called and 3 was added to the end of the array and now the size of the array is 1 and the capacity of the array is 1
The push_back function was called and 0 was added to the end of the array and now the size of the array is 2 and the capacity of the array is 2
The size of this array is 2 and the data in this array are :
3 0
The push_back function was called and 4 was added to the end of the array and now the size of the array is 1 and the capacity of the array is 1
The push_back function was called and 4 was added to the end of the array and now the size of the array is 2 and the capacity of the array is 2
The push_back function was called and 4 was added to the end of the array and now the size of the array is 3 and the capacity of the array is 4
The push_back function was called and 2 was added to the end of the array and now the size of the array is 4 and the capacity of the array is 4
The push_back function was called and 7 was added to the end of the array and now the size of the array is 5 and the capacity of the array is 8
The size of this array is 5 and the data in this array are :
4 4 4 2 7
The push_back function was called and 5 was added to the end of the array and now the size of the array is 1 and the capacity of the array is 1
The push_back function was called and 3 was added to the end of the array and now the size of the array is 2 and the capacity of the array is 2
The push_back function was called and 6 was added to the end of the array and now the size of the array is 3 and the capacity of the array is 4
The push_back function was called and 5 was added to the end of the array and now the size of the array is 4 and the capacity of the array is 4
The push_back function was called and 4 was added to the end of the array and now the size of the array is 5 and the capacity of the array is 8
The push_back function was called and 9 was added to the end of the array and now the size of the array is 6 and the capacity of the array is 8
The size of this array is 6 and the data in this array are :
5 3 6 5 4 9
The copy constructor of the Vec class has been called
The copy constructor of the Vec class has been called
An array with size 5 and capacity of 8 was created
The size of this array was increased to 5 by 0 padding
The destructor of the Vec class was called
The destructor of the Vec class was called
The copy constructor of the Vec class has been called
The copy constructor of the Vec class has been called
An array with size 0 and capacity of 8 was created
The size of this array was increased to 6 by 0 padding
The destructor of the Vec class was called
The destructor of the Vec class was called
The result of the addition of v1 and v2 is stored in v4 and is equal to
The size of this array is 5 and the data in this array are :
7 4 4 2 7
The result of the multiplication of v3 and v2 is stored in v5 and is equal to
The size of this array is 6 and the data in this array are :
20 12 24 10 28 0
The size of v1 is smaller than the size of v3
The size of v4 is smaller than the size of v5
The destructor of the Vec class was called
The destructor of the Vec class was called
The destructor of the Vec class was called
The destructor of the Vec class was called
The destructor of the Vec class was called
garch@ubuntu:~/Desktop/HW3/Q2$
```

شکل 2 – نمونه اجرای تمرین 2

## سوال سوم (C++)

برای تست کد هم از همان فایل Q3main که همراه با تمرین فرستاده شده بود استفاده شد (خطا های syntax این فایل گرفته شده و اجرا شد) .

در این سوال متغیرهای پدر، مادر، همسر و فرزندان که مشخصات فرد نبودند به صورت **public** تعریف شدند (در صورت سوال گفته شده بود که مشخصات شخص از قبیل نام و نام خانوادگی و سن و... به صورت **private** تعریف شوند). همچنین علاوه بر توابع با پیشوند **get** که برای دسترسی به برخی از مشخصات در صورت سوال مشخص شده بود برای دیگر مشخصات هم توابع **get** نوشته شد. همچنین از آنجا که در ادامه نیاز به تغییر مشخصات پیدا می شد توابع با پیشوند **set** تعریف کردیم که از طریق این توابع مشخصات فرد را تغییر دهیم.

در **constructor** کلاس **Human**، هنگام ساخت فرد جدید ابتدا پدر، مادر، همسر و فرزندان آن را با خالی (در پوینترها **nullptr** قرار دادیم) قرار دادیم.

در اپراتورها با توجه به نوع اپراتور شی مورد نظر را **const** قرار دادیم تا از تغییر ناخواسته ی شی در کلاس (برای مثال در اپراتور **>**) جلوگیری کنیم. همچنین از آنجا که کلاس **Human** دارای آرایه ی دینامیک بود، اپراتور **=** را برای آن تعریف کردیم.

در اپراتور **+** از دو **pointer to pointer** استفاده کردیم تا پوینتر فرزندان فعلی را ذخیره کنیم تا در پایان تابع با **delete** کردن فرزندان و دوباره **new** کردن فرزندان این بار به تعداد یکی بیشتر پوینتر فرزندان قبلی را ابتدا در متغیر **Children** والدین ریخته و در آخر پوینتر فرزند جدید را در **Children** بریزیم.

در کلاس **Oracle**، در تابع **marry** اگر شروط برآورده می شد پوینتر هر یک از افراد را در متغیر **spouse** دیگری قرار دادیم. همچنین در تابع **isFamily** برای تشخیص ارتباط بین دو شخص از نام خانوادگی خود شخص و اقوام شخص استفاده کردیم تا تشخیص دهیم آیا دو نفر با هم رابطه ی خانوادگی دارند. در تابع **getFamily** ابتدا همسر و بچه ها را (در صورت وجود) به خانواده اضافه کردیم. سپس خانواده ی همسر را (در صورت وجود اضافه) کردیم. در مرحله ی بعد خانواده ی پدر و بعد هم خانواده ی مادر را به خانواده اضافه کردیم (هر دو در صورت وجود). در مرحله ی آخر هم اگر بچه داشتیم و بچه ها هم خانواده ای داشتند خانواده ی آن ها را به خانواده ی اصلی اضافه کردیم. در آخر هم برای این که در مرحله ی بعد باید تابع **getPopulationOfFamily** را می نوشتیم یک متغیر به نام **populationOfFamily** در کلاس **Oracle** درست کرده و در آخر تابع **getFamily** مقدار **current\_size** (این متغیر در هر مرحله با توجه به تعداد اعضای اضافه شده به خانواده ی اصلی اضافه می شد) را در متغیر **populationOfFamily** ریختیم. در تابع **getPopulationOfFamily** هم ابتدا یک بار تابع **getFamily** را روی شخص مورد نظر اجرا کرده و از آنجا که در انتهای تابع **getFamily** تعداد اعضای خانواده هم در متغیر **populationOfFamily** ذخیره می شود، متغیر **populationOfFamily** را به عنوان خروجی **return** کردیم.

همانطور که در بالا هم اشاره شد برای تست کد نوشته شده از همان فایل **main** فرستاده شده در کانال استفاده شد و فقط اشتباهات **syntax** کد گرفته شد. همچنین در این فایل تست تابع **printChildren** از کلاس **Human** هم اضافه شد.

یک نمونه از اجرای برنامه در شکل 3 آمده است .

```

File Edit View Search Terminal Help
garch@ubuntu:~/Desktop/HW3/Q3$ make
g++ -std=c++17 -Wall -c main.cpp
g++ -std=c++17 -Wall -c -c -o human.o human.cpp
g++ -std=c++17 -Wall -c -c -o oracle.o oracle.cpp
g++ -std=c++17 main.o human.o oracle.o -o main
garch@ubuntu:~/Desktop/HW3/Q3$ ./main
A new person by the name of ALI BAHADORI was created
A new person by the name of BAHAR SHAMS was created
A new person by the name of SHERVIN was created
A new Oracle by the name of SHERVIN was created
ALI and BAHAR can get married
A new person by the name of NEWBORN BABY was created
A new baby has been born to ALI and BAHAR
The gender of the newborn child is female
The haircolor of the newborn child is set to 4
The eyecolor of the newborn child is set to 2
The lastname of the newborn child is set to BAHADORI
The firstname of the person is set to MAHSHID
The age of MAHSHID BAHADORI went up by 1
The children of ALI are
1- MAHSHID BAHADORI
A new person by the name of SHAHIN REZAEI was created
A new person by the name of FARHAD BAHADORI was created
ALI BAHADORI is now the child of SHAHIN REZAEI and FARHAD BAHADORI
MAHSHID BAHADORI and FARHAD BAHADORI are related
the population of the family of SHAHIN REZAEI is: 4
the name of each member of family: ALI BAHAR FARHAD SHAHIN
The destructor of the Oracle class has been called
The destructor of the Human class is called
garch@ubuntu:~/Desktop/HW3/Q3$

```

شکل 3 – نمونه اجرای تمرین 3

## سوال Database

در نوشتن جدول ها نکته ی اول که قابل توجه بود این بود که اگر اسم گذاری همانند نمونه ی داده شد در فایل تمرین باشد یعنی برای مثال Channel را برای کانال داشته باشیم از آنجا که تمام حروف اسم جدول بزرگ و یا تمام حروف آن کوچک نیست ، هر جا که بخواهیم از اسم آن استفاده کنیم باید آن را بین "" قرار بدهیم . نکته ی بعد برای hash کردن پسورد ها بود که برای این کار لازم بود که هر بار که می خواستیم User جدید را اضافه کنید پسورد آن را به تابع md5 می دادیم که این تابع hash پسورد ما را به عنوان خروجی می داد .

در زیر primary key و foreign key های هر جدول آورده شده است .

	Primary key(s)	Foreign key(s)
"User"	id	none
"BlockUser"	blocker_user_id, blocked_user_id	blocker_user_id -> "User" blocked_user_id -> "User"
"Message"	id	sender_id -> "User" receiver_id -> "User"
"Channel"	id	creator_id -> "User"
"Group"	id	creator_id -> "User"
"GroupMessage"	id	group_id -> "Group" sender_id -> "User"
"ChannelMessage"	id	channel_id -> "Channel"

"MessageAttachment"	message_id, attachment_url, attachment_thumb_url	message_id -> "Message"
"GroupMessageAttachment"	message_id, attachment_url, attachment_thumb_url	message_id -> "ChannelMessage"
"ChannelMessageAttachment"	message_id, attachment_url, attachment_thumb_url	message_id -> "GroupMessage"
"ChannelMembership"	user_id, channel_id	user_id -> "User" channel_id -> "Channel"
"GroupMembership"	user_id, group_id	user_id -> "User" group_id -> "Group"

در بخش insert ، برای هر جدول که بعدا باید برای آن query می زدیم به تعداد کافی دستور قرار دادیم . برای مثال برای "User" 10 فرد جدید اضافه کردیم ، برای "Channel" 5 کانال اضافه کردیم برای "ChannelMembership" برای کانال های مختلف تعدادی کاربر اضافه کرده و همچنین بین دو کاربر که بعدا نیاز می شود پیام هایشان را استخراج کنیم 11 پیام فرستادیم .

### جواب سوال پرسیده شده در صورت تمرین

برای این که بتوانیم تعداد بیش از یک ادمین برای کانال ها و یا گروه ها داشته باشیم باید دو جدول به نام های "ChannelAdmins" و "GroupAdmins" اضافه کنیم و در این جدول id گروه ها (کانال ها) و id ادمین ها را ذخیره کنیم همچنین در بخش "ChannelMessage" باید یک ستون دیگر به نام sender\_id اضافه کنیم که بفهمیم کدام یک از ادمین ها پیام را فرستاده است . همچنین در جدول اضافه شده برای ادمین ها باید برای id ادمین به "User" به صورت foreign key باشد و در نهایت در ستون اضافه شده باید sender\_id به primary key ادمین foreign key زده شود .

در 9 شکل زیر خروجی ترمینال دیتابیس برای 9 query خواسته شده قرار داده شده است .

```

id | telegram_id | phone | email | password | first_name | last_name | verification_code | profile_picture_url | created_at | updated_at
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----
4 | a4 | 09120000001 | a4@gmail.com | a717f41c203cb970f96f706e4b12617b | ali4 | alavi4 | 123456798 | http://a4 | 2019-03-31 18:07:08.849247 | 2019-03-31 18:07:08.849247
(1 row)
(END)

```

در query اول مشخصات کاربر با شماره تلفن 09120000001 را گرفته ایم .

id	telegram_id	phone	email	password	first_name	last_name	verification_code	profile_picture_url	created_at	
1	a1	09120000010	a1@gmail.com	827ccb0eea8a706c4c34a16891f84e7b	all8	alav18	123456798	http://a8	2019-04-08 19:09:40.263268	2019-0
2	a2	09120000009	a2@gmail.com	a6bb4faacdff9dcdcb6f6e22bc51eac9	all1	alav11	123456798	http://a2	2019-04-08 19:09:40.302418	2019-0
3	a3	09120000008	a3@gmail.com	4123e5cc43794e5972cee1cf00382445	all3	alav13	123456798	http://a3	2019-04-08 19:09:40.30353	2019-0
4	a5	09120000007	a5@gmail.com	f2d7e2fc28ededdf63c1684a3b6d0c5f	all5	alav15	123456798	http://a5	2019-04-08 19:09:40.305715	2019-0
5	b1	09351000001	b1@gmail.com	f675d8aec99e98b4ee2c6aae5fb12f6b	ben1	bush1	123456798	http://b1	2019-04-08 19:09:40.306586	2019-0
6	b2	09352000002	b2@gmail.com	d68229f521316a03b517e42128c87060	ben2	bush2	123456798	http://b2	2019-04-08 19:09:40.307472	2019-0
7	b3	09353000003	b3@gmail.com	d6128a5c44f82653361a48a08cbc47c2	ben3	bush3	123456798	http://b3	2019-04-08 19:09:40.30839	2019-0
8	b4	09354000004	b4@gmail.com	005d5d1e61effabc44bd74ac51b91d87	ben4	bush4	123456798	http://b4	2019-04-08 19:09:40.309342	2019-0
9	b5	09355000005	b5@gmail.com	37b1fe968daba91ffadbd5a3a9db15	ben5	bush5	123456798	http://b5	2019-04-08 19:09:40.310197	2019-0
10	anlr	09370000001	anlr.jahanshah@gmail.com	44f455185e5ae730f5e12534aaaa5e02	anlr	jahanshah1	123456798	http://anlrj	2019-04-08 19:09:40.311337	2019-0
11	shdt	09380000001	shdtV4RHs@gmail.com	6c0c8a7fd1d9c8124e5ab636376ad492	shdt	V4RHs	123456798	http://shdt	2019-04-08 19:09:40.312321	2019-0
12	a4	09120000001	apstudent2019@gmail.com	a717f41c203cb970f96f706e4b12617b	all4	alav14	123456798	http://a4	2019-04-08 19:09:40.304695	2019-0

در query دوم ایمیل کاربر با شماره تلفن 09210000001 را تغییر دادیم . در شکل بالا جدول “User” چاپ شده است و اگر دقت کنید از آنجا که کاربر ایمیل کاربر مذکور update شده است این کاربر در ته جدول قرار گرفته است در حالی که id آن 4 است . (لازم به ذکر است که باید دقت می شد که هنگام اجرای query باید updated\_at برابر با now() قرار داده می شد تا همانطور که در شکل بالا مشخص است زمان update با زمان create فرق کند)

```
telegram=# select * from "User";
telegram=# select * from "User";
telegram=# select telegram_id from "Channel" where id in
telegram=# (select channel_id from "ChannelMembership" where user_id in
telegram=# (select id from "User" where phone = '09120000001'));
telegram_id
-----
aut_ap_2005
aut_ap_2019
(2 rows)
```

در query سوم نام کانال هایی که کاربر بالا در آنها عضو است بازگردانده شده است .

```
telegram=# select count(id) from "User" where id in
telegram=# (select user_id from "ChannelMembership" where channel_id in
telegram=# (select id from "Channel" where telegram_id = 'aut_ap_2019'));
count
-----
4
(1 row)
```

در query چهارم تعداد کاربر کانال aut\_ap\_2019 بازگردانده شده است .

```
telegram=# select email from "User" where phone like '0935%';
email
-----
b1@gmail.com
b2@gmail.com
b3@gmail.com
b4@gmail.com
b5@gmail.com
(5 rows)
```

در query پنجم ایمیل کاربر هایی که شماره تلفن آن ها با 0935 آغاز می شود باز گردانده شده است .

```

telegram=# select phone from "User" where id in
telegram-# (select blocked_user_id from "BlockUser" where blocker_user_id in
telegram(# (select id from "User" where phone = '09120000001') and
telegram(# ecreated_at > (now() - interval '1 month'));
      phone
-----
 09355000005
 09120000008
(2 rows)

```

در query ششم شماره ی تلفن اشخاصی که توسط فرد با شماره تلفن 09210000001 در یک ماه اخیر بلاک شده اند را استخراج کردیم .

```

telegram=# select email from "User" where id in
(select creator_id from "Channel" where id in
(select channel_id from "ChannelMembership" where user_id in
(select id from "User" where email = 'apstudent2019@gmail.com')
and channel_id in
(select channel_id from
(select channel_id, count(user_id) from "ChannelMembership"
group by channel_id) as subq
where count > 3)));
      email
-----
 a5@gmail.com
 b1@gmail.com
(2 rows)

```

در query هفتم ایمیل صاحبان کانال هایی که اعضای آن بیشتر از ۳ عضو دارد و کاربر با ایمیل [apstudent2019@gmail.com](mailto:apstudent2019@gmail.com) نیز در آن ها عضو است را برگرداندیم . در این query در بخشی از query که می خواستیم تعداد اعضا را مشخص کنیم باید یک با یک subquery جدول جدیدی می ساختیم که نیاز بود برای این subquery اسم جدید گذاشته شود که این اسم به صورت “subq” در کد مشخص است .

```

telegram=# select phone from "User" where id in
telegram-# (select creator_id from "Channel" where id in
telegram(# (select channel_id from "ChannelMessage" where
telegram(# updated_at in (select max from
telegram(# (select max(updated_at) from
telegram(# "ChannelMessage" group by channel_id) as subq
telegram(# where max < (now() - interval '1 month'))));
      phone
-----
 09351000001
(1 row)

```

در query نهم هم شماره تلفن کاربران صاحب کانال هایی را برگردانید که در یک ماه اخیر هیچ پستی در کانال آن ها گذاشته نشده است بازگرداندیم .



```

telegram=# select message_type, message_text from "Message" where
telegram-# (sender_id = (select id from "User" where telegram_id = 'amir.jahanshahi')
telegram-# and receiver_id = (select id from "User" where telegram_id = 'shDiV4RHs') or
telegram-# (receiver_id = (select id from "User" where telegram_id = 'amir.jahanshahi')
telegram-# and sender_id = (select id from "User" where telegram_id = 'shDiV4RHs'))
telegram-# order by created_at desc limit 10;
message_type | message_text
-----+-----
text         | Bye!
text         | Alright Imma go!
text         | You should rest!
text         | Hang in there!
text         | OH NO!
text         | Nothing just tired!
text         | So why the attitude
text         | Oh! alright.
text         | Woah! Calm down, just wanted to say hi
text         | Hey back! what do you want
(10 rows)

```

در query هشتم 10 پیام آخر مکالمه ی دونفره ی بین دو کاربر با آی دی های تلگرامی shDiV4RHs و amir.jahanshahi را برگرداندیم .

توجه : نظر به این که نحوه ی **export** کردن **database** در کلاس توضیح داده نشد و همچنین تلاش های من برای **export** کردن آن بی نتیجه ماند ، ازمحتوای هر جدول (با اجرای دستور **select \* from** یک **screenshot** گرفته و عکس ها را در گزارش ضمیمه کردم .

## بارگذاری در Git

برای بارگذاری کد ها در **github** ، هنگام ساخت **repository** در قسمت آپشن ها فایل **.gitignore** را با تنظیمات **C++** تشکیل دادیم که این باعث شد فایل های با پسوند نامطلوب خود به خود به این فایل اضافه شده و ما در ادامه فقط نام فایل **main**

را به آن اضافه کردیم . سپس در لینوکس دستور **git init** را اجرا کرده و سپس با دستور **git clone** و بعد از آن **git remote -v** در یک دایرکتوری دلخواه **repository** را **clone** کردیم . در ادامه فایل های پوشه های برنامه های نوشته شده را در بخش **clone** شده اضافه کرده و با استفاده از دستور

”folder“ git add آن پوشه را در repository اضافه کردیم . سپس با دستور ”comment“ –m git commit فایل ها را برای بارگذاری آماده کردیم . در نهایت هم با دستور git push origin master فایل ها را فرستادیم . لازم به ذکر است که اگر نیاز به تغییرات در هر یک از فایل های commit شده بود تنها نیاز بود که فایل را ویرایش کرده و سپس دوباره آنرا add سپس commit و در نهایت push کنیم و فایل مورد نظر در repository خود به خود به روز رسانی می شد .

در شکل صفحه ی بعد می توانید مراحل انجام کار را مشاهده کنید .

```
File Edit View Search Terminal Help
garch@ubuntu:~/Github/AP-HW3$ git add Q2
garch@ubuntu:~/Github/AP-HW3$ git add Q3
garch@ubuntu:~/Github/AP-HW3$ git add DB
garch@ubuntu:~/Github/AP-HW3$ git add .gitignore
garch@ubuntu:~/Github/AP-HW3$ git add README.md
garch@ubuntu:~/Github/AP-HW3$ git commit -m "second commit"
git: 'commit' is not a git command. See 'git --help'.

The most similar command is
commit
garch@ubuntu:~/Github/AP-HW3$ git commit -m "second commit"
[master 5346efb] second commit
25 files changed, 1825 insertions(+)
create mode 100644 DB/Tablescreenshots/BlockUser.png
create mode 100644 DB/Tablescreenshots/Channel.png
create mode 100644 DB/Tablescreenshots/ChannelMembership.png
create mode 100644 DB/Tablescreenshots/ChannelMessage.png
create mode 100644 DB/Tablescreenshots/ChannelMessageAttachment.png
create mode 100644 DB/Tablescreenshots/Group.png
create mode 100644 DB/Tablescreenshots/GroupMembership.png
create mode 100644 DB/Tablescreenshots/GroupMessage.png
create mode 100644 DB/Tablescreenshots/GroupMessageAttachment.png
create mode 100644 DB/Tablescreenshots/Message.png
create mode 100644 DB/Tablescreenshots/MessageAttachment.png
create mode 100644 DB/Tablescreenshots/User.png
create mode 100644 DB/create.sql
create mode 100644 DB/insert.sql
create mode 100644 DB/query.sql
create mode 100644 Q2/Makefile
create mode 100644 Q2/main.cpp
create mode 100644 Q2/vec.cpp
create mode 100644 Q2/vec.h
create mode 100644 Q3/Makefile
create mode 100644 Q3/human.cpp
create mode 100644 Q3/human.h
create mode 100644 Q3/main.cpp
create mode 100644 Q3/oracle.cpp
create mode 100644 Q3/oracle.h
garch@ubuntu:~/Github/AP-HW3$ git push origin master
Username for 'https://github.com': MSalehG
Password for 'https://MSalehG@github.com':
Enumerating objects: 32, done.
Counting objects: 100% (32/32), done.
Compressing objects: 100% (31/31), done.
Writing objects: 100% (31/31), 1.47 MiB | 250.00 KiB/s, done.
Total 31 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/MSalehG/AP-HW3
62b85ab..5346efb master -> master
garch@ubuntu:~/Github/AP-HW3$
```

دستورات اجرا شده برای بارگذاری در git

همچنین با کلیک روی لینک زیر می توانید به repository ساخته شده در git بروید .

<https://github.com/MSalehG/AP-HW3>