گزارش پروژه ی مبانی برنامه سازی

دانشکده مهندسی کامپیوتر

نیم سال اول ۹۸-۹۹

روزبه پیراعیادی

شماره دانشجویی :۹۸۱۰۱۳۰۶

در این گزارش توضیحاتی در رابطه با کلاینت و سرور پروژه ی چت اپلیکیشن خواهد آمد.

۱. کلاینت

۱-۱ تابع choice_getter

```
int choice_getter(char [][20], int , char[] );
```

تابع choice_getter تابعی است که گزینه ی انتخاب شده توسط کاربر در منو های مختلف را برمی‌گرداند.این تابع یک آرایه ی دو بعدی از گزینه های مختلف ، تعداد آن ها و یک آرایه به عنوان تیتر گزینه ها گرفته سپس تا زمان زده شدن کلید space گزینه ها و تیتر را چاپ کرده و با فشرده شدن کلید های بالا و پایین مکان نشانگر(-->) را عوض می کند.توجه کنید که برای آنکه بتوانیم کاراکتر های آخر(خانه های ۱۸،۱۷ و ۱۶)را به نشانگر تبدیل کنیم باید \0 را به خانه ی ۱۹ منتقل کنیم.

۱-۲ تابع account_menu

```
void account_menu();
```

تابع account_menu تابعی بدون ورودی و خروجی است که مربوط به منوی اولیه ی برنامه می باشد.این تابع به کمک تابع socket_maker سوکتی برای ارتباط با سرور ساخته و به کمک choice_getter با کاربر ارتباط برقرار می کند.در این تابع کاربر مجاز به انتخاب Register، Login و Exit است.

۱-۳ تابع main_menu

```
void main menu(char []);
```

تابع main_menu تابعی بدون خروجی است که به عنوان تنها ورودی خود یک token می گیرد که در ادامه برای ارسال ریکوئست به سرور از آن استفاده می کند.این تابع هنگامی که در تابع account_menu

عمل Login به درستی انجام گیرد فراخوانی خواهد شد. در این تابع نیز کاربر به کمک تابع choice_getter می تواند از بین گزینه های Create Channel، Join Channel و Logout انتخاب کند.

۴-۱ تابع chat_menu

```
void chat_menu(char []);
```

تابع chat_menu نیز تابعی بدون خروجی است که به عنوان تنها ورودی خود یک token گرفته و در ادامه برای ارسال ریکوئست به سرور از آن استفاده می کند. این تابع هنگامی که در تابع main_menu عمل Login به درستی انجام گیرد فراخوانی خواهد شد. در این تابع نیز کاربر به کمک تابع choice_getter می تواند از بین گزینه های Send Message،Refresh، Channel Members و Leave Channel انتخاب کند.

۵-۱ نکات مهم:

```
    printf("press any key to continue\n");
    getch();
```

- توجه کنید که چون تابع choice_getter به سرعت کل صفحه را پاک می کند برای آنکه کاربر بتواند پاسخ را ببیند لازم است ()getch کنیم.

```
closesocket(client_socket);
```

- توجه داشته باشید که در هر مرحله پس از برقراری ارتباط باید سوکت را ببندیم.

- همچنین توجه کنید که در انتهای هر یک از تابع های account_menu، main_menu و chat_menu خود آنها فراخوانی می شوند تا در صورت رخ دادن خطا یا زمان های مانند register که به منو ی دیگری منتقل نمی شویم در همان منو بمانیم.

۲. سرور

۲-۱ استراکت user

```
typedef struct
{
    char username [max_username]  ;
    char token [max_token]  ;
    char channel [max_channel_name]  ;
    int index_message ;
}user ;
```

از این استراکت برای ذخیره ی اطلاعات مربوط به کاربران آنلاین استفاده می‌شود.در این استراکت یوزرنیم کاربر، توکن او ،نام کانالی که عضو آن است و شماره ی آخرین پیامی که دیده است ذخیره می گردد.

۲-۲ متغیر where_to_write

این متغیر به ما نشان می دهد باید در کجای آرایه ی online_users بنویسیم.

۲-۳ تابع token_maker

```
char * tokenmaker ();
```

این تابع تابعی است بدون ورودی که به صورت رندوم یک توکن تولید می کند و آن را برمی گرداند.

۲-۴ تابع find_token

```
int find_token(char []);
```

این تابع توکنی را که در ورودی دریافت کرده را در آرایه ی گلوبال مربوط به کاربران جستجو می کند.در صورت یافتن کاربری با این توکن شماره او در آرایه و در صورت پیدا نشدن کاربری با آن توکن 1- را بر می گرداند.

۵-۲ تابع socket_maker

```
void socket_maker();
```

از این تابع برای برقراری ارتباط استفاده می شود.توجه کنید که متغیر های server_socket و client_socket گلوبال هستند در نتیجه این تابع نه خروجی دارد نه ورودی.

۶-۲ تابع send_respond

```
void send_respond(cJSON * , int);
```

این تابع بدون خروجی یک جیسون و یک عدد صحیح گرفته و به کمک تابع send آن جیسون را برای کلاینت ارسال می کند.توجه کنید که آن عدد صحیح حداکثر سایز جیسون را نشان می دهد و در برنامه از دو ماکرو ی normal_respond و huge_respond استفاده شده است.

۷-۲ تابع is_word_in_message

```
int is_word_in_message (char [] , char[]);
```

این تابع به عنوان ورودی اول یک پیام و به عنوان ورودی دوم یک کلمه می گیرد.سپس در صورت وجود داشتن آن کلمه در پیام ۱ و در غیر این صورت ۰ برمی گرداند.

۳. رنگ و شکل

۱-۳ تابع set_color

```
void set_color(int);
```

از این تابع برای تغییر رنگ صفحه استفاده می شود و تنها ورودی آن که یک عدد صحیح است مشخص می کند صفحه به چه رنگی در می آید.

۲-۳ تابع dash

```
void dash ();
```

این تابع تعدادی دش چاپ می کند.

۳-۳ تابع ce

```
void ce();
```

این تابع نماد ce را چاپ می کند. (:

۴. کتابخانه ی My_cJSON

۱-۴ استراکت cJSON

در این استراکت یک پوینتر به آیتم بعد ، یکی به آیتم قبل و یکی هم به فرزند وجود دارد.همچنین یک متغیر type وجود دارد که نوع آیتم را مشخص می کند.(1- برای آیتم،0 برای آبجکت و ۱ برای آرایه) همچنین دو پوینتر به char هم وجود دارد که یکی اسم آیتم و دیگری رشته ی ذخیره شده در آن را نشان می دهد.

۲-۴ تابع cJSON_CreateItem

```
cJSON* cJSON_CreateItem()
```

این تابع مقدرای فضا برای یک آیتم و همین طور اسم و رشته ی مربوط به آن اختصاص می دهد.(توجه کنید که اسم و رشته پوینتر هستند)

۳-۴ تابع cJSON_CreateObject

```
cJSON* cJSON CreateObject()
```

این تابع مقدرای فضا برای یک آبجکت و همین طور اسم و رشته ی مربوط به آن اختصاص می دهد.

۴-۴ تابع cJSON_CreateArray

```
cJSON * cJSON CreateArray()
```

این تابع مقدرای فضا برای یک آرایه و همین طور اسم و رشته ی مربوط به آن اختصاص می دهد.

۵-۴ تابع cJSON_CreateString

```
cJSON* cJSON CreateString(char * string)
```

این تابع یک رشته به عنوان ورودی گرفته آیتمی ساخته ، رشته را درون آیتم قرار داده و پوینتر آن را بر می گرداند.

۶-۴ تابع cJSON_AddItemToObject

```
void cJSON_AddItemToObject (cJSON * first , char * name , cJSON * need_to_be_added )
```

یک آیتم و نام در نظر گرفته شده برای آن را می گیرد و به یک آبجکت اضافه می کند.

۷-۴ تابع cJSON_Print

```
char *  cJSON_Print (cJSON * json)
```

یک پوینتر به جیسون گرفته و هر چه باشد چاپ شده ی آن را بر می گرداند.(در صورت لزوم cJSON_PrintArray را صدا می زند)

۸-۴ تابع cJSON_PrintArray

```
char * cJSON_PrintArray(cJSON * json)
```

همان طور که در قسمت قبل توضیح داده شد یک آرایه را به رشته تبدیل کرده و برمی گرداند.

۹-۴ تابع cJSON_Parse

```
cJSON * cJSON_Parse(char * string )
```

این تابع برعکس تابع cJSON_Print عمل کرده و برای رشته ای که می گیرد شروع به ساختن جیسون کرده و سپس ریشه(first) را برمی گرداند و برای این کار از تابع put_in_json کمک می گیرد.

۱۰-۴ تابع put_in_json

```
char * put_in_json (char * buffer , cJSON * json)
```

تابع put_in_json یک رشته و یک پوینتر به جیسون گرفته سپس یک نام و یک رشته می خواند و در جیسون می ریزد.اما اگر به کاراکتر] بر بخورد تابع cJSON_ParseArray را صدا خواهد کرد.

۱۱-۴ تابع cJSON_ParseArray

```
char * cJSON_ParseArray(char *string , cJSON * json)
```

این تابع بسیار شبیه به تابع cJSON_Parse عمل می کند با این تفاوت که برای آرایه است و خودش جیسون را نمی سازد بلکه آن را به عنوان ورودی می گیرد.

۱۲-۴ تابع cJSON_GetObjectItem

```
cJSON * cJSON_GetObjectItem(cJSON * first  , char *string )
```

این تابع یک آبجکت و یک رشته گرفته و اشاره گری به آیتمی با آن نام را برمی گرداند.

۱۳-۴ تابع cJSON_GetArrayItem

```
cJSON* cJSON_GetArrayItem (cJSON * array , int n)
```

این تابع یک آرایه و یک عدد صحیح گرفته و اشاره گری به خانه ای با شماره ی آن عدد را برمی گرداند.

۱۴-۴ تابع cJSON_GetArraySize

```
int cJSON GetArraySize (cJSON * array)
```

این تابع یک آرایه گرفته و با حرکت روی آبجکت ها تا به NULL برسد سایز آرایه را یافته و برمی گرداند.

۱۵-۴ تابع cJSON_AddItemToArray

```
void cJSON_AddItemToArray (cJSON * array , cJSON * need_to_be_added )
```

این تابع یک آبجکت را به آرایه ای که از ورودی می گیرد می چسباند.

پایان