تمرین کامپیوتری سوم مبانی کامپیوتر و برنامه سازی 🖥️

اساتید درس: دکتر هاشمی، دکتر مرادی
طراحان: مجید صادقی نژاد، آریا عازم ، مهدی حاجی

- **یک مورد امتیازی جدید، تحت عنوان ارسال پیام (sendMessage) با طول نامعلوم (بیش از ۲۵۶ کاراکتر) اضافه شد.**

مقدمه 💥

سلام، در تمرین کامپیوتری سوم قصد داریم، یک **پیام رسان ساده** طراحی کنیم. در این تمرین کامپیوتری با مفاهیمی همچون Linked List ، حافظه ی پویا، struct و ... کار خواهیم کرد.
همچنین توصیه ی اکید می گردد در جلسه ی توجیهی این تمرین کامپیوتری شرکت کنید تا حداکثر آشنایی با برنامه را به دست آورید !

اهداف آموزشی 🖊️

- ☐ استفاده از لیست های پیوندی به نحو مناسب
- ☐ استفاده از توابع برای افزایش خوانایی و سرعت و کارایی کد
- ☐ به کارگیری حافظه ی پویا و آزادسازی آن
- ☐ طراحی struct های مناسب برای طراحی برنامه

طراحی برنامه

برنامه ما از چندین موجودیت[1] تشکیل شده است که به صورت زیر هستند :

• کاربر (User)

- هر کاربر دارای یک Username است که در سیستم ما یکتا[2]ست، یعنی دو کاربر نمیتوانند username یکسان داشته باشند. همچنین در نام کاربری تنها استفاده از کاراکتر های ساده انگلیسی مجازند و استفاده از کاراکتر هایی مانند فاصله ، حروف فارسی و ... مجاز نیست. (حداکثر ۵۰ کاراکتر)

[1] Entity
[2] Unique

- یک رمز ورود که برای ورود کاربر به سیستم مورد بررسی قرار میگیرد. در رمز عبور تنها استفاده از کاراکتر های ساده انگلیسی مجازند و استفاده از کاراکتر هایی مانند فاصله ، حروف فارسی و ... مجاز نیست. (حداکثر ۵۰ کاراکتر)
- هر کاربر یک لیست از Chat ها دارد که چت های کاربر با کاربران دیگر در این لیست ذخیره می گردد .

- **چت (Chat)**

- هر چت دارای یک نام (Name) می باشد. کاربر در هنگام ساخت یک چت جدید این نام را مشخص می کند. (حداکثر ۵۰ کاراکتر)
- هر چت دقیقا بین دو کاربر ساخته می شود که این دو کاربر میتوانند با یکدیگر در این چت ارتباط برقرار کنند.
- هر چت شامل لیست پیام های (Messages) رد و بدل شده بین دو کاربر می باشد که از قدیم به جدید نگهداری می شوند. همچنین در هر چت، فقط دو کاربر همان چت، امکان ارسال و خواندن پیام ها را دارند.

- **پیام (Message)**

- هر پیام حداکثر 256 کاراکتر خواهد داشت. (دریافت پیام با طول نامعلوم (بیش از ۲۵۶ کاراکتر) امتیازی خواهد بود)
- هر پیام دقیقا یک نویسنده (User) دارد.
- هر پیام دقیقا متعلق به یک چت (Chat) است.

- **سیستم اصلی (System)**

- این موجودیت ، دربرگیرنده ی تمام اطلاعات کلی سیستم ماست :
  - لیست تمام یوزرها
  - کاربر لاگین شده (در صورت عدم وجود باید این مقدار null باشد)
  - چت انتخاب شده[3](در صورت عدم وجود باید این مقدار null باشد)

توضیحات کلی (مربوط به هر دو فاز)

برای هر یک از دستورات برنامه، تعدادی نمونه ی ورودی و خروجی نوشته شده است که برای هر یک از این نمونه ها باید توجه داشته باشید که :

[3] Current selected chat

- تمام ورودی ها و خروجی ها باید دقیقا مطابق نمونه ها باشند. تصحیح و تحویل برنامه شما به وسیله ماشین صورت خواهد گرفت بنابراین در صورت وجود کوچتکرین تفاوت بین خروجی خواسته شده و خروجی برنامه شما، ماشین از برنامه ی شما ایراد خواهد گرفت.
- در نمونه های ورودی بخش های قرمز رنگ بخش هایی هستند که توسط برنامه چاپ می شود و بخش های سیاه توسط کاربر نوشته می شوند. (نیازی به پیاده سازی رنگ ها نیست ! رنگ های فقط برای راهنمایی بیشتر اضافه شده اند!)
- تضمین میشود که حالاتی خارج از مثال های آورده شده ، هنگام تحویل پروژه از شما خواسته نمی شود.

این پروژه برای راحتی شما به دو فاز تقسیم بنده شده است :

- کارکرد[4] های توضیح داده شده در بخش فاز اول حتما باید در هنگام تحویل فاز اول انجام شده باشند.
- **همچین تمامی طراحی struct ها (مربوط به موجودیت های برنامه) نیز مربوط به فاز اول می باشد تا ساختار برنامه ی شما از همان ابتدا به طرز صحیحی شکل گیرد.**
- قابلیت های فاز دوم به فاز اول اضافه می شوند و فاز دوم تکمیل کننده ی فاز اول است .
- برنامه شما باید در پایان فاز دوم به طور کامل اجرا گردد.

نکته مهم : تعداد کاربران، چت ها و تعداد پیام ها در هر چت باید نامحدود فرض شود. بنابراین شما **نباید** از آرایه برای نگهداری کاربر، چت و پیام ها استفاده کنید و برای نگه داری این موجودیت[5] ها **باید فقط و فقط از لیست های پیوندی**[6]استفاده کنید!

[4] Functionality
[5] Entity
[6] Linked list

امکانات و نحوه ی استفاده ی برنامه (فاز اول)

دریافت کننده دستور

- پس از شروع برنامه، عبارت Enter Command نمایش داده خواهد شد و منتظر دریافت ورودی از کاربر می ماند.

نمونه ورودی
`Enter Command:`

- کاربر میتواند یکی دستوراتی که در زیر آورده شده اند را وارد کند و در صورتی که دستور کاربر، نیاز به دریافت آرگومان داشته (برای مثال دستور ثبت نام نیاز به آرگومان نام کاربری و رمز عبور دارد)، در خطوط بعدی، در هر خط یک آرگومان از کاربر دریافت می گردد (در بخش توضیحات دستور ها و نمونه ها این توضیحات همراه با مثال داده شده است!) و پس از دریافت تمام آرگومان ها، دستور متناسب با توضیحات آن که در بخش زیر آورده شده است ، اجرا می گردد. پس از اجرای دستور مجددا عبارت Enter Command چاپ می شود و منتظر دریافت ورودی از کاربر می ماند این فرایند تا هنگامی که کاربر ورودی exit را وارد نکند ادامه خواهد داشت
- توجه کنید که ممکن است دستوری ورودی خاصی نداشته باشد (مانند exit) که در این صورت پس از ورود دستور ، دستور ما اجرا می گردد .
- همچنین تضمین میشود که در حین ورودی گرفتن یک دستور ، دستور دیگری توسط کاربر وارد نمی گردد.
- باید توجه کنید که برنامه ما باید در برابر دستورات نامعتبر، رفتار مناسبی ارائه دهد که توضیحات آن در ادامه آمده است.
- دقت داشته باشید که برای کارایی هرچه بهتر کد برای تعریف قالب دستورات حتما از قابلیت Define در c استفاده نمایید . توجه کنید که Define متفاوت از متغیر گلوبال است چرا که شما قادر به تغییر آن در طول برنامه نیستید ! شکل زیر صورت قابل قبول است :

```
#define LOGIN "login"
#define SIGNUP "signup"
#define LOGOUT "logout"
```

• ثبت نام (signup)

هنگامی که یک فرد جدید قصد دارد به سیستم اضافه شود، پس از نمایش پیام دریافت کننده ی دستور (Enter Command) ، دستور signup را وارد می کند. پس از وارد کردن این دستور توسط کاربر در خط بعد متن Username به همراه دو نقطه، چاپ می شود و منتظر دریافت ورودی از کاربر می ماند پس از این که کاربر ورودی خود را وارد کرد و کلید enter را فشرد در خط بعد متن Password به همراه دو نقطه چاپ می شود و منتظر ورودی کاربر می ماند و پس از آن که کاربر ورودی خود را وارد کرد و enter را زد، پردازش ثبت نام آغاز می شود و پیام مناسب چاپ می گردد.

دقت کنید که نحوه ی دریافت آرگومان هایی مانند username , password برای سایر دستورات که در ادامه توضیح داده خواهد شد نیز به همین صورت است.

- با توجه به اینکه نام کاربری در سیستم ما باید یکتا[7] باشد ، در ابتدا چک می کنیم که نام کاربری قبلا در سیستم ثبت نشده باشد. در صورتی که نام کاربری از پیش در سیستم وجود داشت پیام مناسب ، دقیقا مشابه آنچه در مثال ها آمده در ترمینال چاپ شود.
- در صورتی که ثبت نام موفقیت آمیز بود کاربری با مشخصات وارد شده در سیستم ما ثبت می شود و پیام مناسب ، دقیقا مشابه آنچه در نمونه ها آمده در ترمینال چاپ شود.
- دقت داشته باشید که در صورتی که ثبت نام موفقیت آمیز بود ، کاربر نیاز است که login کند و فقط صرف ثبت نام جدید موجب لاگین شدن در سیستم نمیشود !
- تضمین می شود کاربر هنگامی که در سیستم لاگین کرده است اقدام به ثبت نام نمی کند.

نمونه ورودی اول

```
Enter Command: signup
Username: majid
Password: 1234
```

نمونه خروجی اول

```
User majid successfully registered!
```

نمونه ورودی دوم

```
Enter Command: signup
Username: repeatedOne
Password: 1234
```

نمونه خروجی دوم

```
Error: Username repeatedOne already exists.
```

[7] Unique

• ورود (login)

در سیستم ما ممکن است که کاربری قبلا ثبت نام کرده باشد و سعی داشته باشد که صفحه شخصی خود دسترسی پیدا کند .

- در این صورت ما کاربر با وارد کردن نام کاربری و رمز عبور ، در صورت صحیح بودن هر دو ، باید بتواند وارد شود و در کنسول پیام مناسب ، دقیقا مشابه مثال ها چاپ شود .
- در صورتی که اطلاعات وارد شده صحیح نبود ، کاربر را با پیام مناسب در کنسول متوجه موضوع کنیم.
- در صورتی که کاربری از قبل لاگین کرده بود و خارج نشده بود ، و دستور login صدا زده شد، باید بدون گرفتن نام کاربری و رمز ، به کاربر گفته شود که یک کاربر باز[8] داریم و نمی توان لاگین کرد.

نمونه ورودی اول

```
Enter Command: login
Username: majid
Password: 1234
```

نمونه خروجی اول

```
User majid logged in successfully!
```

نمونه ورودی دوم

```
Enter Command: login
Username: majid
Password: 7890
```

نمونه خروجی دوم

```
Error: Invalid username or password.
```

نمونه ورودی سوم: در صورت وجود یک کاربر از قبل لاگین شده

```
Enter Command: login
```

نمونه خروجی سوم

```
Error: There is an open session, please logout first.
```

[8] session

• خروج (logout)

در صورتی که یک نفر وارد سیستم شده بود از قبل امکان اینکه مجددا کاربر جدیدی (یا همان کاربر قبلی) وارد سیستم بشنود نیست ، برای همین حتما باید ابتدا خارج شوند و سپس مجددا لاگین کنند.

- در هر لحظه از برنامه که کاربر دستور Logout را وارد کرد ، در صورتی که کاربری لاگین شده بود از سیستم ما خارج میشود .
- در صورتی که کاربری login نکرده بود ، سیستم ما دقیقا مشابه مثال ها خطا می دهد.

نمونه ورودی اول

```
Enter Command: logout
```

نمونه خروجی اول

```
User majid logged out successfully!
```

نمونه ورودی دوم

```
Enter Command: logout
```

نمونه خروجی دوم

```
Error: No user is currently logged in.
```

• خروج از برنامه (exit)

در صورتی که کاربر (چه لاگین شده ، چه لاگین نشده) این پیام را به عنوان دستور ورودی داد، باید برنامه را خاتمه دهیم. دقت کنید که در پس از خاتمه ی برنامه شما حتما موظف به آزاد[9] کردن حافظه های استفاده شده هستید . در صورتی که این کار را نکنید و پس از خاتمه برنامه ، حافظه هایی که در برنامه اشغال شده بودند را آزاد نکرده باشید و با نشتی حافظه [10]مواجه شده باشید از شما نمره کسر خواهد شد.

نمونه ورودی

```
Enter Command: exit
```

[9] Free
[10] Memory Leak

امکانات و نحوه ی استفاده ی برنامه (فاز دوم)

• ساختن چت جدید (newChat)

- در صورتی که کاربری لاگین نکرده بود خطای مناسب نمایش داده شود. (دقیقا مشابه مثال ها)
- در صورتی که username کاربری که قصد ایجاد چت با او را داریم در سیستم وجود نداشت خطا نمایش داده شود. (دقیقا مشابه مثال ها)
- در صورتی که کاربر مقصد چت ، خود یوزری که لاگین کرده بود ، باشد خطا نمایش داده شود(دقیقا مشابه مثال ها)
- اگر چتی میان دو کاربر مورد نظر دقیقا با این نام وجود داشت ، به کاربر نمایش داده شود. (دقیقا مشابه مثال ها)

در صورتی که هیچ یک از خطاهای بالا رخ نمیداد ، کاربر میتواند یک چت با نام مذکور با کاربر مقصد بسازد و سپس باید در ترمینال به کاربر موفقیت آمیز بودن عملیات را اعلام نماییم.

نمونه ورودی اول

```
Enter Command: newChat
Enter chat name: ThirdProjectForICSP
Enter selected username: mahdi
```

نمونه خروجی اول

```
Error: No user is currently logged in.
```

نمونه ورودی دوم

```
Enter Command: newChat
Enter chat name: ThirdProjectForICSP
Enter selected username: mahdi
```

نمونه خروجی دوم

```
Error: User mahdi not found.
```

نمونه ورودی سوم

```
Enter Command: newChat
Enter chat name: ThirdProjectForICSP
Enter selected username: mahdi
```

نمونه خروجی سوم

```
Error: You cannot create a chat with yourself.
```

نمونه ورودی چهارم

```
Enter Command: newChat
Enter chat name: ThirdProjectForICSP
Enter selected username: mahdi
```

نمونه خروجی چهارم

```
Error: Chat with name ThirdProjectForICSP already exists.
```

نمونه ورودی پنجم

```
Enter Command: newChat
Enter chat name: ThirdProjectForICSP
Enter selected username: mahdi
```

نمونه خروجی پنجم

```
Chat with the name ThirdProjectForICSP created between mahdi and
majid.
```

• انتخاب یک چت (selectChat)

درصورتی که کاربری لاگین کرده باشد و چتی برای کاربر لاگین شده با نام ذکر شده وجود داشته باشد ، باید آن چت را به عنوان چت انتخاب شده [11] در system انتخاب کنیم .

- ممکن است که چتی با نام ذکر شده برای کاربر لاگین شده وجود نداشته باشد که باید خطای مناسب دقیقا مشابه مثال ها به کاربر داده شود
- ممکن است که کاربری لاگین نکرده باشد که در این صورت هم باید خطای مناسب دقیقا مشابه مثال ها آورده شود
- نکته : وقتی یک چت انتخاب شد تا زمانی که انتخاب جدیدی صورت نگیرد همه دستورهای بعدی مثل showChat و یا sendMessage برای همان چت عمل خواهد کرد.

نمونه ورودی اول

```
Enter Command: selectChat
Enter chat name: ExistedChatName
```

نمونه خروجی اول

```
Chat ExistedChatName has been selected as the current chat
```

نمونه ورودی دوم

```
Enter Command: selectChat
Enter chat name: ExistedChatName
```

نمونه خروجی دوم

```
Error: No user is currently logged in.
```

نمونه ورودی سوم

```
Enter Command: selectChat
Enter chat name: ExistedChatName
```

نمونه خروجی سوم

```
Error: Chat ExistedChatName not found for user majid.
```

[11] current_chat

• ارسال پیام (sendMessage)

در صورتی که کاربر لاگین کرده بود و یک چت از میان چت های خود را انتخاب[12] کرده بود ، باید بتواند که در آن چت یک پیام برای کاربر مقصد ارسال کند. در صورتی که ارسال ما موفقیت آمیز بود ، پیامی لازم نیست در کنسول چاپ شود .

- اما در صورتی که چتی به عنوان چت انتخاب شده[13] در سیستم[14] وجود نداشت ، باید کاربر را با خطای مناسب و بدون گرفتن پیام ورودی از این موضوع آگاه کنیم. (دقیقا مشابه مثال ها)

نمونه ورودی اول

```
Enter Command: sendMessage
Receiver: Mahdi
Enter message: salam Mahdi, khobi? project sevom mabani ro didi?
```

نمونه خروجی اول

نمونه ورودی دوم

```
Enter Command: sendMessage
```

نمونه خروجی دوم

```
Error: No chat selected. Please select a chat first.
```

• نمایش پیام های چت (showChat)

ممکن است که کاربر لاگین کرده بخواهد که تمام پیام های یک چت را ببیند ، در این صورت باید ابتدا با استفاده از دستور selectChat که در بالا توضیح داده شده است ، یک چت را انتخاب[15] کند ، سپس می تواند با این دستور ، تمام پیام های ارسال شده در چت را به همراه نویسنده هرپیام مشابه نماید . در ابتدای این پیام ها باید نام چتی که در حال نمایش پیام های آن هستیم آورده شود .

- در صورتی که چتی به عنوان چت انتخاب شده [16] وجود نداشت ، خطای مناسب ارسال شود.

---

[12] Select Chat
[13] Selected Chat
[14] System
[15] Select
[16] Selected chat

نمونه ورودی اول

```
Enter Command: showChat
```

نمونه خروجی اول

```
Messages in Chat 'ICSP-projects':
Majid: Salam Mahdi , Khobi ?
Mahdi: Salam , ghorbanat Majid . To chetori?
Majid: Shokr , manam khobam , nafasi miad va mire.
Mahdi: Project sevom mabani ro didi ?
Majid: Mesle Ab khordane baram :)
```

نمونه ورودی دوم

```
Enter Command: showChat
```

نمونه خروجی دوم

```
Error: No chat selected. Please select a chat first.
```

- ## دستورات غیر مجاز

در صورتی که کاربر (چه لاگین شده ، چه لاگین نشده) دستوری خارج از دستورات بالا را به عنوان دستور ورودی وارد کرد ، باید با خطای مناسب که در مثال آمده است ، او را از این موضوع آگاه کنیم .

نمونه ورودی

```
Enter Command: SomeRamdomWrongCommand
```

نمونه خروجی

```
Error: Unknown command.
```

بخش امتیازی 🎉

• دستور لود کردن اولیه کاربران ها در سیستم (loadUsers)

این دستور آدرس و نام یک فایل را به عنوان ورودی دریافت میکند که در این فایل چندین نام کاربری و رمز عبور در هر خط مشابه مثال زیر وجود دارد . که در هر خط یک نام کاربری و یک رمز عبور وجود دارد که این دو با استفاده از یک فاصله [17]از یکدیگر جدا شده اند:

نمونه اطلاعات درون فایل ورودی

```
Username1 Password1
Username2 Password2
Username3 Password3
Username4 Password4
```

پس از خواندن فایل مورد نظر باید تمام کاربران درون فایل در صورتی که نام کاربری و رمز عبور آنها معتبر باشد ، در سیستم ساخته شده باشند و امکان لاگین کردن با آن نام کاربری و رمز عبور و باقی دستورات در سیستم برای این کاربران موجود باشد .

- در صورتی که در یک خط اشکالی وجود داشت باید به کاربر گفته شود که خط شماره ی x مشکلی داشته و از آن خط فایل چشم پوشی[18] شود .(باقی خط ها در صورتی که معتبر بودند باید خوانده شوند)(همچنین ممکن است که چندین خط از فایل این دارای فرمت بد باشند)
- در صورتی که فایل به هر دلیلی باز نشد یا اصلا وجود نداشت مشابه مثال ها خطای مناسب داده شود.
- در صورتی که اضافه کردن کاربران موفقیت آمیز بود ، پیام مناسب همراه با این که چند کاربر به سیستم اضافه شده ان مشابه مثال اضافه گردد .
- همچنین دقت کنید که در پایان عملیات حتما فایل بسته[19] شود.
- تضمین میشود که نام کاربری های درون فایل تکراری نیستند و قبلا در سیستم ثبت نشده اند.

نمونه ورودی اول

```
Enter Command: loadUsers
Enter filename:  myfile.txt
```

---

[17] Space
[18] Skip
[19] closed

نمونه خروجی اول

```
Error: Unable to open file myfile.txt.
```

نمونه ورودی دوم

```
Enter Command: loadUsers
Enter filename:  myfile.txt
```

نمونه خروجی دوم

```
Users successfully loaded from myfile.txt , 4 user(s) added.
```

نمونه ورودی سوم

```
Enter Command: loadUsers
Enter filename:  myfile.txt
```

نمونه خروجی سوم

```
Warning: Invalid format in line 2. Skipping.
Warning: Invalid format in line 4. Skipping.
Users successfully loaded from myfile.txt , 6 user(s) added.
```

- ## نوشتن برنامه به صورت چند فایلی (Multi-File)

شما باید موجودیت های و توابع مربوطه به آن ها را به بخش های مختلف بشکنید و در فایل های مختلف c. قرار دهید تا ساختار برنامه مرتب تر و خوانا تر شود همچنین برای این کار نیاز است تا فایل های h. نیز ایجاد شوند. نحوه تقسیم بندی برنامه بر اساس فایل های به عهده ی خودتان می باشد.

نکات مهم 🙂

- استفاده از متغیرهای Global به هیچ وجه مجاز نمی باشد و در صورت نیاز باید داده ها را در قالب آرگومان به توابع انتقال دهید.
- در صورت وجود هرگونه سوالی، می توانید سوالات خود را در گروه تلگرامی درس مطرح نمایید تا باقی دوستان هم از سوال و جواب های مطرح شده در گروه درس استفاده کنند .
- فایل های برنامه ی خود را در یک فایل فشرده با فرمت "zip" و با نام "CA3-SID.zip" قرار دهید که SID همان شماره ی دانشجویی شماست. برای مثال اگر شماره ی دانشجویی شما 810103555 باشد باید نام فایل خود را "CA3-810103555.zip" قرار دهید و آن را در قسمت در نظر گرفته شده در صفحه درس در سامانه ایلرن آپلود نمایید.

- فایل zip شما باید فقط شامل فایل های .h , c باشد همچنین **نباید** آن ها را در یک پوشه قرار دهید و سپس فشرده کنید . تمامی فایل ها را مستقیما فشرده کنید
- برنامه‌های شما باید با زبان برنامه‌نویسی C نوشته شود و استفاده از دیگر زبان‌های برنامه‌نویسی مجاز نیست.
- **مهلت تحویل فاز اول این تمرین تا ساعت ۲۳:۵۹ روز ۱۶ دی ماه است**.
- **مهلت تحویل فاز دوم این تمرین تا ساعت ۲۳:۵۹ روز ۲۶ دی ماه است**.
- شما **در طول این ترم** چهار روز مهلت اضافه برای تمارین کامپیوتری دارید که می توانید آن ها را بین تمارین کامپیوتری به انتخاب خودتان تقسیم کنید در صورتی که از مهلت اضافی خود برای یک تمرین استفاده کنید می توانید آن تمرین را بدون کسر نمره، با تاخیر تحویل دهید. همچنین **توجه داشته باشید که تمارین کامپیوتری با بیش از دو روز تاخیر تحویل گرفته نمی شوند.**
- تمرین های کامپیوتری برای یادگیری برنامه نویسي و مباحث مطرح شده در کلاس طراحی می‌شوند و انجام آن‌ها به صورت انفرادی خواهد بود. همچنین در صورت شباهت میان دو پروژه (که به وسیله ی نرم افزارهای مربوطه چک می‌شود.) برای هر دو نفر نمره صفر در نظر گرفته خواهد شد.

نحوه ی نمره دهی 😁

- ☐ نام گذاری مناسب متغیر ها، توابع و فایل ها (۱۵ نمره)
- ☐ مرتب بودن و تمیزی کد به همراه کامنت گذاری مناسب (۳۰ نمره)
- ☐ عدم وجود خطای کامپایل، اخطار (۲۰ نمره)
- ☐ کارایی و عملکرد برنامه به همراه تسلط بر کد نوشته شده (۳۵ نمره)
- ☐ بخش امتیازی - خواندن از فایل (۱۰ نمره)
- ☐ بخش امتیازی - چند فایلی[20] (۱۰ نمره)
- ☐ بخش امتیازی - دریافت ورودی با اندازه ی نامعلوم (۵ نمره)

موفق باشید 🙂

[20] Multi-File