



شرط

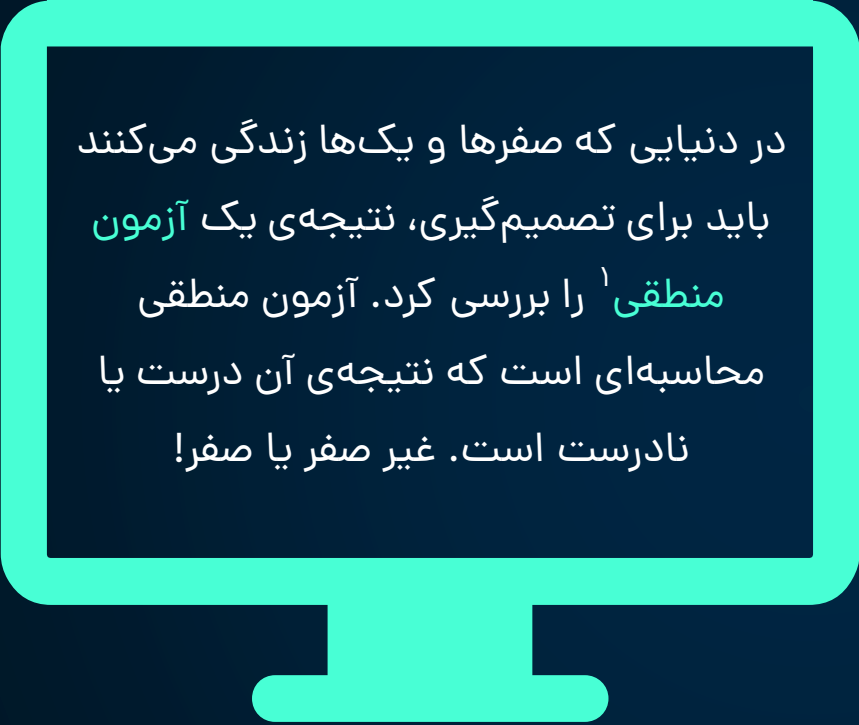
جلسه پنجم

# بسم الله الرحمن الرحيم

## Condition



کارگاه مبانی برنامه نویسی - دانشکده مهندسی کامپیوتر دانشگاه امیرکبیر



در دنیایی که صفرها و یک‌ها زندگی می‌کنند  
باید برای تصمیم‌گیری، نتیجه‌ی یک **آزمون**  
**منطقی**<sup>۱</sup> را بررسی کرد. آزمون منطقی  
محاسبه‌ای است که نتیجه‌ی آن درست یا  
نادرست است. غیر صفر یا صفر!

ما هر روز برای انجام هر یک از کارهایمان در  
حال تصمیم‌گیری هستیم. مثل این که  
انتخاب کنیم چه غذایی بخوریم و یا آیا لازم  
هست برای خروج از خانه لباس گرم بپوشیم یا  
نه؟ همه‌ی این تصمیم‌گیری‌ها بر اساس  
شرایط خاصی به وجود می‌آیند که با عوض  
شدن اوضاع ممکن است تصمیم ما هم تغییر  
پیدا کند. اهمیت شرط در زندگی کاملاً مشهود  
است و نیازی به بازگو کردن آن نیست، پس  
باید ببینیم چطور می‌توانیم شرط‌های مورد  
نظرمان را در دنیای کامپیوتر اعمال کنیم.

# فهرست



# سوال اول: یک مثال مثلثات



برنامه‌ای بنویسید که سینوس و علامت کسینوس یک زاویه را دریافت کند، کسینوس آن را محاسبه کرده و سپس بگوید زاویه در کدام ربع دایره‌ی مثلثاتی واقع شده.



آیا به خاطر دارید که در جلسه‌ی قبل گفتیم فرض می‌کنیم همه‌ی ورودی‌های برنامه کاملاً منطقی و درست هستند چون هنوز توانایی لازم برای محدود کردن ورودی‌ها را نداریم؟



به نظر شما ممکن است این کد با چه خطاهایی مواجه شود؟



برای مثال، آیا سینوس می‌تواند مقدار ۵ را داشته باشد؟ اگر این عدد را وارد کنید برنامه با چه مشکلی روبه‌رو می‌شود؟

سعی کنید ورودی‌های نادرست را به کمک شرط‌ها محدود کنید.



# امتیازی ★



در این بخش می‌توانید به سراغ کد ماشین حساب جلسه‌ی قبل بروید که آن را با کمک کد خدا و Botfather دیباگ کرده بودید. حال سعی کنید که ماشین حساب خود را ارتقا دهید و خواسته‌ی سوال را به عنوان یکی از بخش‌های جدید ماشین حساب تصور کنید تا قابلیتی به ماشین حساب شما افزوده شود.



توجه: به نوع ورودی‌های قبلی ماشین حساب دقت کنید. برنامه در بخش‌های قبلی همیشه دو عدد اعشاری یا صحیح را دریافت می‌کرد اما در این سوال ورودی دوم شما از نوع کاراکتر است. از چه راه‌هایی می‌توانید این مشکل را برطرف کنید؟

## سوال دوم: نقطه شرط

در ذهن خود درس هندسه را یک مرور سریع کنید!



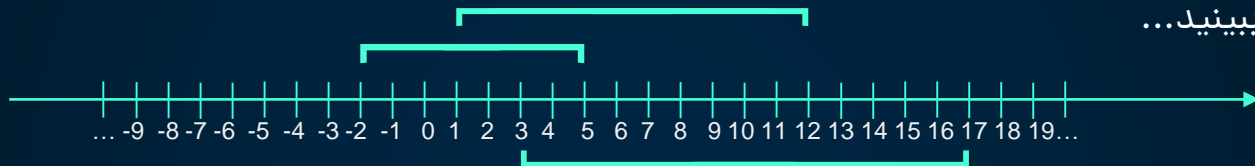
در نگاهی گذرا هم می‌توانید ببینید که در این درس از شرط‌های زیادی استفاده می‌کنیم؛ مثلاً اگر شیب دو خط برابر باشد، می‌گوییم این دو خط موازی هستند. ما می‌توانیم با برنامه‌نویسی، بسیاری از همین شرط‌ها را با سرعت و دقت بیش‌تر چک کنیم تا به نتیجه برسیم.



برنامه‌ای بنویسید که مختصات طول و عرض ۴ نقطه را به عنوان ورودی بگیرد و اگر همه‌ی این نقاط روی یک خط باشند، عبارت "All in one line"، اگر سه نقطه روی یک خط باشند، عبارت "Three in one line"، اگر این ۴ نقطه یک لوزی را تشکیل می‌دهند، عبارت "Diamond" را در خروجی چاپ کند و اگر هیچ یک از شرط‌های بالا درست نبود عبارت "None" را در خروجی چاپ کند.

# سوال سوم: کمتر هم می‌شه؟!

شکل زیر را ببینید...



بازه‌ای که بین هر سه بخش هم‌پوشانی دارد را بیابید. چطور این کار را کردید؟ حال همان شرطهایی را که در

ذهن خود مرور کردید تا به بخش مشترک بین سه بازه برسید را به کامپیوترتان نیز آموزش دهید!

برای این کار برنامه‌ی شما باید ابتدا سه بازه از اعداد را دریافت کند. این سه بازه در خطوط متوالی داده می‌شود

و در هر خط عدد اول و آخر هر بازه داده می‌شود. حال شما می‌خواهید با کمک شرطها و مطالب قبلی‌ای که

آموخته‌اید ببینید که اشتراک این بازه‌ها چه اعدادی هستند.

نکته‌ی مهم برنامه‌ی شما این است: این کار را با کمترین تعداد شرط ممکن انجام دهید. آیا برنامه‌ای که نوشتید

با تعداد شرط کمتر هم برقرار می‌شود؟

## سوال چهارم: زنگ تفریح

قطعه کد زیر را اجرا کنید و در مورد علت هر خروجی‌ای که مشاهده می‌شود، بحث کنید (به جز خط آخر. خط آخر را فقط ببینید 😊) در آینده با این کاراکتر بیشتر آشنا خواهید شد.



```
printf("true = %d\tfalse = %d\n", true, false);
if (3 * 4)
    printf ("Result of 3 * 4 is: %d\n", 3 * 4);
if (3 * 4 == 12)
    printf ("Result of 3 * 4 == 12 is: %d\n", 3 * 4 == 12);
if ('b')
    printf ("ASCII code of 'b' is: %d\n", 'b');
if (' ')
    printf ("What is space ASCII code?\n");
if ('\0')
    printf ("****the ASCII code of \0 is 0****\n");
printf ("You will learn more about \0 in \"String\". BOOOM!");
```



گذاشتن `stdbool.h` در ابتدای برنامه برای شناختن `true` و `false` فراموش نشود.



به یاد دارید در ابتدای دستورکار گفتیم شرطها نتیجه‌ی یک آزمون منطقی را چک می‌کنند و این نتیجه می‌تواند صفر باشد یا غیر صفر؟



اینجا دیدید با وجودی که باید ۰ معادل نادرست و ۱ معادل درست باشد، اینطور نیست و `if` هر عددی غیر از صفر را درست در نظر می‌گیرد.



# و اما سوال آخر: به یاد بازی های کودکانه



بابا می‌خوایم بازی کنیم نمی‌شه که خشک صحبت کنیم.

همون دستورکار قبلی کافی بود. من نمی‌تونم کتابی حرف بزنم.



باشه باشه قبول کردم دیگه همون عادی صحبت کنیم.



بریم بچه‌ها منتظرن... برو شروع کن.

سلام و خدا قوت به شماهایی که تا این‌جای دستورکار رو انجام دادین. حالا سوال آخر رو با هم پیش می‌بریم که نشون بدیم حتی بازی‌های بچگانه رو هم می‌تونیم به کامپیوترمون یاد بدیم.



از شما می‌خوایم قبل از هر کاری، کد ارسالی رو بررسی کنین. فکر می‌کنین برنامه‌ای که نوشته شده (با این‌که الان ناقصه و قراره که کاملش کنیم) داره قوانین کدوم بازی رو می‌گه؟





همونطور که احتمالا حدس زدین این پروژه دقیقا همون بازی سنگ کاغذ قیچیه... پس می‌خوایم با کامپیوتر سنگ کاغذ قیچی بازی کنیم :)

چطوری؟

برای شروع بازی اول باید با هم به توافق برسین که برنده‌ی بازی باید چند امتیاز بگیره؟ یعنی شرط خاتمه‌ی بازی رو لازمه که کامپیوتر بدون وگرنه تا جان در بدن داشته باشه باید با اون بازی کنین (=)



قدم دوم چیه؟ یه دستتون رو ببرین پشتتون یه دست رو کنار صورتتون نگه دارین تا تعداد دفعاتی که برنده می‌شین رو بشمرین.

حالا کامپیوتر وارد بازی می‌شه. متاسفانه یا خوشبختانه دستی برای این سبک بازی نداره و بازی براش یه مدل دیگه تعریف شده :)

پس دست و بالتونو جمع کنین و بذارین رو کیبورد :دی



اذیت نکن ملت رو :|



خب حالا دوباره به کد برگردین و سعی کنین توضیح بدین که کامپیوتر قراره با شما چطور بازی کنه. به نکته‌ی خوب اینه که دیگه لازم نیست تعداد دفعات برنده‌شدن‌تون رو بشمرین، چون حریف هوشمندتون بدون اشتباه این کار رو براتون انجام می‌ده. فقط کافیه سنگ کاغذ یا قیچی رو با انتخاب یک عدد مشخص کنین. حریف هم از تابع رندوم رو برای این انتخاب استفاده می‌کنه.



قبل از بازی کد رو کامل کنید تا کامپیوتر (این رقیب و داور شریف) با هر دور بازی، امتیاز برنده رو افزایش بده.



امیدواریم که از این بازی لذت برده باشین. تا کارگاه بعدی خدا نگهدار همگی \*-\*

;