توی این جلسه struct و نحوه تعریف تابع رو برای اون یاد می‌گیریم.

struct چیست!؟

احتمالا تا حالا واستون پیش اومده که وسط نوشتن یک برنامه نیاز داشته باشید یک سری اجزا و خواص اونها رو به صورت یکجا نگه داری کنید، مثلا اطلاعات تعدادی شهر(مثل مساحت، جمعیت و ....)

این نیاز وقتی پررنگ تر میشه که قراره روی این اشیا عملیات انجام بدیم. مثلا شهرهامونو بر اساس جمعیتشون مرتب (sort) کنیم.

در این صورت اگه struct بلد نبوده باشید احتمالا به مشکل برخوردید!

نحوه تعریف یکstruct

برای تعریف یک struct از الگوی زیر استفاده میکنیم:

```
struct city{
...
};
int main(){
...
}
```

از همین‌جا به این نکته توجه می‌کنیم که اون ; آخر کار خیلی مهمه! و اگه فراموش بشه ممکنه به ارورهای مختلف برخورد کنید و سر اشکال‌زدایی برنامه حسابی گیج بشید.

خواص struct داخل قسمت ... تعریف میشوند:

```
struct city{
int Population;
double Area;
};
```

خواصی که به sturct نسبت میدید از هر نوعی میتونه باشه (مثل..,int, char, double)

همونطوری که بقیه متغیرها تا حالا تعریف میشدن میتونیم یک نمونه از struct هم داشته باشیم:

```
int main(){
city Tehran;
return 0;
}
```

برای تغییر یکی از خواص شئ مورد نظر مثل زیر رفتار میکنیم:

```
Tehran.Population = 7*1e6
//7*1e6 = 7000000
```

حالا میتونید هرکاری که با بقیه متغیرها میکردید با struct خودتون هم انجام بدید! مثلا میتونید یک city به عنوان ورودی به یک تابع خاص بدید:

```
void MyFunc{city a, city b){
...
}
```

توی اون قسمت ... که اول کار بهش اشاره کردیم هم میشه تابع تعریف کرد:

```
double PerCapita(){
    return this->Area / this->Population;
}
```

الان هروقت Tehran.PerCapita() رو صدا بزنیم به ما سرانه زمین به ازای هر نفر توی شهر تهران رو برمیگردونه.this توی تابع یعنی خواصیت Area مربوط به همین شهر(تهران) که تابع برای اون صدا زده شده مورد نظره!

برای شئ‌هاتون میتونید عملگر هایی مثل > و + و... هم تعریف کنید! به دلیل کاربرد بیشتر عملگر > مثال زده میشه:

```
bool operator < (const city &a, const city &b){
if (a.Population == b.Population)
return a.Area < b.Area;
return a.Population < b.Population;
}
```

استفاده از const و & برای اینه که مطمئن باشیم در طول مقایسه دو شئ اطلاعات مربوط به اونا هیچ تغییری نمیکنن و صرفا با هم مقایسه میشن!

به این ترتیب دو شهر a و b اول بر حسب جمعیت و در صورت تساوی بر حسب مساحت مقایسه میشن!

حالا که برای struct ما عملگر > تعریف شده میتونیم از تابع ()sort برای مرتب کردن چند شهر استفاده کنیم یا شهرهامون رو داخل یک ()set نگه داریم!

الان دیگه میتونیم مشکل ابتدای کار(مرتب کردن تعدادی شئ بر اساس یک خاصیت) رو حل کنیم. پس میتونید به عنوان تمرین برنامه ای بنویسید که ابعاد تعدادی مکعب رو از ورودی بگیره و توی سه سطر شماره ورودی مکعب‌ها رو بر اساس افزایش طول، عرض و ارتفاع(هرکدوم توی یک سطر) چاپ کنید .

آخر کار لازمه یه بار دیگه یادآوری کنم که آخر تعریف یک struct هرگز ; فراموش نشه...