جلسه ی چهارم (توابع و برنامه نویسی ساخت یافته)

اوایلی که برنامه نویسی به وجود آمد باید برنامه به صورت یک قطعه بزرگ نوشته میشد که باعث میشد بعضی مجموع دستورات چندین بار تکرار بشوند بعلاوه خطا یابی وتغییر وفهم برنامه خیلی مشکل بود.

بنابراین قرار بر این شد که بجای تعداد زیادی(چند ده هزار)خط دستور پشت سرهم برنامه را به تعدادی برنامه های کوچک مجزا با ورودی خروجی های مشخص ایجاد نمایند تا برنامه اصلی با توجه به نیازها با دادن ورودی مناسب و اجرا دیگر برنامه ها اهداف برنامه را تامین کند نتیجه این که تعدادی برنامه کوچک داریم که خطا یابی و فهمشان بدلیل کوچک بودنشان آسان است.

```
}(لیست پارامترها)نام تابع    نوع مقدار بازگشتی تابع
مجموعا دستورات
}
```

شکل کلی توابع بصورت بالا می باشد حالا یه مثال می زنم که هیچی مثل مثال شفاف کننده نیست

```
#include<iostream.h>
#include<conio.h>
void star(){
cout<<"*******";
}
int main(){
cout<<"this is main"<<endl;
star();
getch();
return 0;
}
```

در این مثال نوع باز گشتی تابع star بدون نوع یعنی void تعریف شده چرا که این تابع قرار نیست چیزی بعنوان خروجی به برنامه بر گرداند لیست پارامتر ها هم خالیست چرا که این تابع نیازی به ورودی هم ندارد.

خب حالا می خوایم برنامه رو کامپایل کنیم

اول تابع main رو پیدا میکنیم بعد اجراش می کنیم خط اول روی مانیتور عبارت This is main را چاپ می کنیم و نشانه گر مانیتور رو به خط بعد می بریم بعد خط بعد رو اجرا می کنیم که برای این کارمجبوریم تابع star را پیدا کنیم و اجرا کنیم بعد از اجرا آن به خط بعدی تابع main برمیگردیم یعنی خط سوم بقیش را هم خودتان میدانید

خب حالا فرض کنید می خواهیم تابع star تعداد مشخصی ستاره چاپ کند پس تابع ما ورودی می خواهد. یعنی پارامتر می خواهد حالا مثال را با دقت بخوانید.

```
#include<iostream.h>
#include<conio.h>
void star(int n){
int i;
for(i=0;i<n;I++)
   cout<<"*";
}
int main(){
int a;
star(3);
cin>>a;
```

cplearning.blogfa.com

```
star(a);
getch();
return 0;
}
```

یک تابع می تواند چند پارامتر داشته باشد

```
void star(int a,int b,float a)
```

تابع در صورتی که نوع بازگشتی داشته باشد به صورت زیر استفاده می شود

```
#include
#include
int fabs(int a){
if(a<0)a=-a;
return a;
}
int main(){
int n;
cin>>n;
n=fabs(n);
cout<<n;
getch();
return 0;
}
```

در این مثال تابع قدر مطلق تعریف شده می بینید که مقدار باز گشتی با دستور returnبه برنامه بر گردانده شده برای گرفتن این مقدار باز گشتی باید عبارت فراخوانی کننده تابع را برابر یک متغیر مناسب قرارداد در واقع نام تابع حاوی مقدار باز گشتی است. با این روش باز گشت مقادیر از تابع فقط می توان یک مقدار را بعنوان نتیجه از تابع گرفت اما روش هایی برای بازگشت چند مقدار از تابع وجود دارد که بعدا ذکر خواهد شد.

اما در مورد متغیرهای عمومی و محلی. به متغیر هایی که درون لیست پارامترها یا درون توابع تعریف می شوند متغیر های محلی گویند این متغیر ها فقط از درون توابع خودشان قابل دسترسی هستند و توابع دیگر نمی توانند مقادیر متغیر های محلی تابع دیگری را تغییر دهند.البته میشود موقع فراخوانی تابع به متغییرهایی که در لیست پارامترها تعریف شده اند مقدار اولیه داد. با توجه به مفاهیم ذکر شده توابع مختلف می توانند متغییر های محلی هم نام داشته باشند که هیچ دخلی هم با هم ندارند و هر کدام کار خود را می کنند. متغییر های محلی بعد از پایان اجرای تابع از حافظه حذف می شوند.

اما متغییر های عمومی متغیر هایی هستند که خارج از توابع ومعموما قبل از تعریف تابع mainتعریف می شوند این متغیر ها در سراسر برنامه قابل دسترسی اند اگردریک تابع یک متغیر محلی هم نام با متغییر عمومی تعریف شود متغییر عمومی باروش های معمولی قابل دسترس نخواهد بود و از آن به بعد متغییر محلی بدون هیچ گونه دخالتی از جانب متغییر عمومی کار خود را می کند برای دست یابی به متغییر عمومی در تابع مذکور در چنین وضعی اگر فرض کنیم نام متغییر aباشد باید بجای aبنویسیم a::در واقع a به تنهایی یعنی متغییر محلی وa::یعنی متغییر عمومی

کلاسهای حافظه

تمام متغیر ها دارای دو ویژگی هستند

1.حوزه ی متغیر

2.طول عمر متغیر

coplearning.blogfa.com

حوزه ی متغییر می گوید که متغییر در چه جاهایی از برنامه قابل دسترسی است
مثلا متغیر های محلی فقط درون تابعی که درآن تعریف شده اند قابل دست یابی اند
همچنین طول عمر یک متغیر محلی از زمان اجرا تابع شروع می شود و با خاتمه اجرا تابع تمام میشود

انواع کلاسهای حافظه

کلاس حافظهautomatic

تمام متغیر های محلی که تا کنون تعریف می کردیم از این کلاس میباشند

کلاس حافظهstatic

این کلاس دارای دونوع استاتیک محلی واستاتیک عمومی می باشد
تفاوت متغیر های استاتیک محلی با متغیرمحلی معمولی در این است که بعد از پایان اجرا تابع از بین نمیروند و مقدار خود را برای اجرا بعدی تابع حفظ میکنند
تفاوت متغیر استاتیک عمومی با متغیر عمومی در حوزه ی متغییر است
متغییر عمومی در تمام برنامه قابل دستیابی است اما متغیر استاتیک عمومی فقط در توابعی که بعد از تعریف متغیر مذکور تعریف شده اند قابل دستیابی اند
نکته مهم دیگر این که متغیر های استاتیک چه عمومی وچه محلی مقدار اولیه ی صفر دارند بر خلاف دیگر متغیر ها.

کلاس حافظه یexternی

تمام متغیر های عمومی(متغیر هایی که خارج از توابع تعریف می شوند) عضو این کلاس اند
این متغیر ها در تمام زمان اجرا برنامه وجود دارند و در تمام برنامه قابل دستیابی اند

کلاس حافظه register

این کلاس حافظه به کامپایلر پیشنهاد می کند متغیر در ثبات های CPUایجاد شوند این کار باعث افزایش سرعت کار بامتغیر مذکور میشود چرا که سیستم عامل مجبور نیست برای کار با آن به ثبات ها منتقلش کند ودوباره نتایج را به RAMبرگرداند.
از آنجا که CPUثبات های کوچک ومحدودی دارد این کلاس حافظه محدودیت هایی دارد
1.فقط برای متغیر های محلی قابل استفاده است
2.فقط برای انوع صحیح و کاراکتری واشاره گرها قابل تعریف است
3.نمی توان اشاره گری به متغیری با این کلاس تعریف کرد
4.دستور استفاده از این کلاس حالت پیشنهاد داردو اگر CPUنتواند ثباتی در اختیار برنامه قرار دهد متغیر در RAMایجاد می شود.

```
register int a;
static char c;
```

معمولا کلاس هایautomaticوexternرا برای تعریف متغیر نمی نویسند
البته نوشتنشان خوانایی برنامه را افزایش میدهد
خب فقط یک نکته مانده که چون خارج از کتاب هست و بیانش هم مشکل هست بی خیال میشم

تا به حال توابع را قبل از تابع mainتعریف می کردیم اما می توانیم الگوی توابع را قبل از تابع mainتعریف کنیم وبعد از تابع mainخود توابع را تعریف کنیم به مثال دقت کنید

```
#include<iostream.h>
```

cpplearning.blogfa.com

```
#include<conio.h>
int fabs(int a);     الگوی تابع
int main(){
int n;
cin>>n;
n=fabs(n);
cout<<n;
getch();
return 0;
}
int fabs(int a){    خود تابع
if(a<0)a=-a;
return a;
}
```

همچنین در الگوی توابع برای پارامترها می توانید نام پارامتر را ننویسید و فقط نوع آن را ذکر کنید چرا که در الگوی تابع کامپایلر فقط می خواهد بداند تابع چند پارامتر واز چه نوعی دارد.

جلسه ی بعد راجع به اشاره گرها و مر جع ها واین که با استفاده از آنها چگونه تابعی میتواند از طریق پارامتر ها مقادیری به برنامه بر گرداند.

cpplearning.blogfa.com