

به نام خالق هستی بخش

راهنمای کار با هدر فایل File.h

با سلام، در این بخش به شما هدر فایلی را معرفی خواهیم کرد که بتوانید به سادگی بر روی فایل ها عملیات ورودی و خروجی انجام دهید. مثلاً یک فایل متنی ایجاد کنید اطلاعاتی را بر روی آن بنویسید و یا از آن بخوانید.

تذکر مهم: ما این فایل ها را بر اساس کامپایلر های gcc نوشته ایم پس انتظار نداشته باشید با مایکروسافت ویژوال استادیو نتایج مطلوب بگیرید.

ابتدا فایل File.h را کنار فایل اصلی خود قرار داده و آن را include کنید:

```
#include "File.h"
```

این فایل شامل `#include "iostream"` هم چنین دستور `using namespace std;` می شود و لازم نیست در فایل خود این دستورات را بنویسید اگر چه نوشتن آن ها اشکالی ایجاد نمی کند! خوب به سراغ معرفی کتابخانه می رویم:

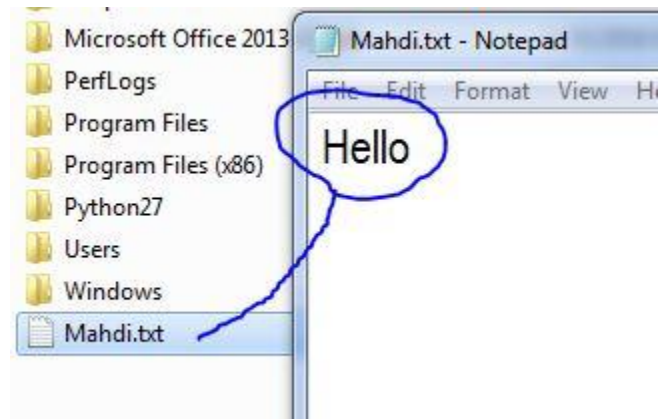
اولین کار تنظیم مسیر است!!:

اولین کاری که باید انجام دهید تنظیم مسیر است دقت کنید اگر می خواهید فایل ایجاد کنید، انجام این کار موجب ایجاد فایل نخواهد شد ولی اگر می خواهید از فایل که وجود دارد اطلاعاتی را دریافت کنید با این کار مسیر به کلاس File معرفی شده و کلاس آماده خواندن از فایل با توابعی (یا بهتر بگوییم متدها) که بعداً معرفی می کنیم خواهد بود:

```
File.SetPath("C:\\Mahdi.txt");
```

با این دستور مسیر `C:\Mahdi.txt` در نظر گرفته می شود اگر این فایل وجود داشته باشد می توانید اطلاعاتی را به آن اضافه کنید و یا از آن اطلاعاتی را بخوانید و اگر فایلی با این نام در مسیر مربوطه (ریشه درایو C¹) وجود نداشته باشد، با اولین دستور ورودی در فایل این فایل ایجاد خواهد شد.

فرض کنیم فایل با محتوای "Hello" وجود داشته باشد:



¹ فرهنگستان ادب فارسی در واژه گزینی معادل رانه را برای واژه درایو در نظر گرفته!!

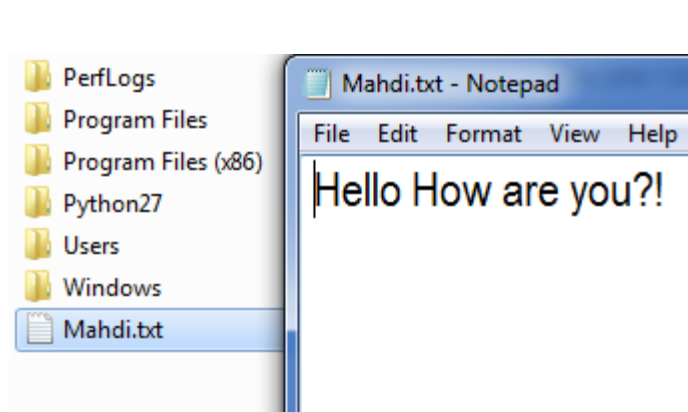
حال پس از دستوری که در قبل برای تنظیم آدرس در نظر گرفتیم
دستور زیر را اضافه نمایید:

```
File.Write(" How are You?!");
```

برنامه را کامپایل کرده و اجرا کنید برنامه شما چیزی شبیه باید چیزی
شبیه به تصویر زیر باشد:

```
1  #include "iostream"
2  #include "File.h"
3
4  using namespace std;
5
6  int main(){
7
8
9      File.SetPath("C:\\Mahdi.txt");
10
11     File.Write(" How are you?!");
12
13     return 0;
14 }
```

پس از اجرای برنامه به مسیری که فایل وجود دارد بروید و نگاهی به
آن بیندازید:



ملاحظه کردید که بسادگی توانسید یک فایل متنی را ویرایش کنید!

نکات ریز :

1- همانطور که می بینید در تعیین مسیر فایل بجای C:\Mahdi.txt از C:\\Mahdi.txt استفاده کرده ایم اما چرا؟ خیلی واضح است قطعا می دانید که برخی از علائم در ++C یا دیگر زبان ها با نماد \ همراه کاراکتر می آیند ممکن است غیر چاپی یا چاپی باشد مثل n\ که برای ایجاد خط جدید بوده و غیر چاپی است و یا \" که برای نوشتن " داخل متن است نماد \\ هم برای وارد نمودن \ داخل رشته می باشد.

2- متنی که به تابع Write() به عنوان ورودی داده ایم را با یک فاصله شروع کردیم که این کار برای مرتب سازی متون داخل فایل متنی است زیرا اگر فاصله را رعایت نمی کردایم H در کلمه How به O در کلمه Hello می چسبید، که ظاهر خوشایندی نداشت!

همانطور که دیدید تابع `Write()` بدون اینکه خط را رد کند در همان خط که متن وجود دارد عملیات چاپ را انجام می دهد حال اگر بخواهیم متن چاپ شود و خط رد شود می توانیم از تابع `WriteLine()` استفاده کنیم. برای امتحان این موضوع تابع `WriteLine` را با `Write()` در برنامه قبل جایگزین کرده و سپس مجدداً با تابع `Write()` متنی را اضافه کنید ملاحظه می کنید که

تابع `WriteLine()` پس از چاپ متن را خط را رد می کند و دستور `Write()` متن را در خط بعدی چاپ می کند!

تا اینجا هستید بگذارید شما را با تابع `Clear()` هم آشنا کنیم!:

در هر مرحله ای از کار که بودید می توانید با استفاده از تابع `Clear()` متن موجود در فایل متنی را پاک کنید تنها کافی است از دستور `File.Clear();` استفاده کنید در این صورت تمام محتوای فایلی که در `SetPath()` مشخص کرده اید پاک خواهد شد.

آخرین نکته ای هم که در این مبحث باید به آن اشاره کنیم ایجاد فایل است همانطور که گفته شد با اجرای دستور `SetPath` و مقدار دادن به آن چنان چه آن فایل از قبل وجود داشته باشد با دستور نوشتن در فایل مقادیر به آن اضافه می شود ولی اگر چنین فایلی وجود نداشته باشد با اولین دستور `Write()` یا `WriteLine()` و نوشتن محتوایی در آن فایل مورد نظر در مسیری که شما اشاره کرده اید ایجاد می شود.

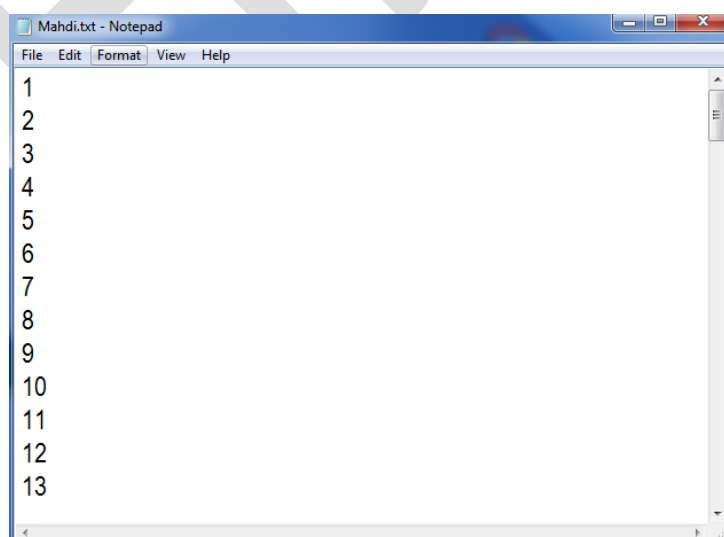
خوب مبحث نوشتن در فایل را با یک مثال پایان می دهیم:

مثال : برنامه ای بنویسید که فایلی با نام Mahdi.txt را در کنار فایل اصلی ایجاد کرده و اعداد یک تا صد را به ترتیب در هر خط چاپ کند:

متن اصلی برنامه :

```
1  #include "iostream"
2  #include "File.h"
3
4  using namespace std;
5
6  int main(){
7
8
9      File.SetPath("Mahdi.txt");
10
11     for(int i = 1 ; i <= 100 ; i++)
12         File.WriteLine(i);
13
14     return 0;
15 }
```

نتیجه اجرا:



خواندن از فایل های متنی:

برای خواندن از فایل های متنی پس از تنظیم آدرس فایل با تابع
SetPath() باید از توابع مربوط به ورودی استفاده کنید که آن ها عبارت
اند از:

ReadChar()-1

ReadWord()-2

تابع ReadChar()

با استفاده از این تابع قادر خواهید بود به اندازه یک کاراکتر از فایل
خود بخوانید در واقع با هر بار فراخوانی این تابع یک کاراکتر از فایل
خوانده شده و برگشت داده می شود به مثال پایین دقت کنید:

فرض کنیم فایلی با نام Mahdi.txt در مسیر C:\Mahdi.txt قرار دارد
که داخل آن عبارات زیر نوشته شده است:

"In the name of God

I'm Mahdi Moradi

I glad to write a HeaderFile for you that can make
programming easy.

Believe Your ability!"

حال برنامه را به صورت زیر می نویسیم:

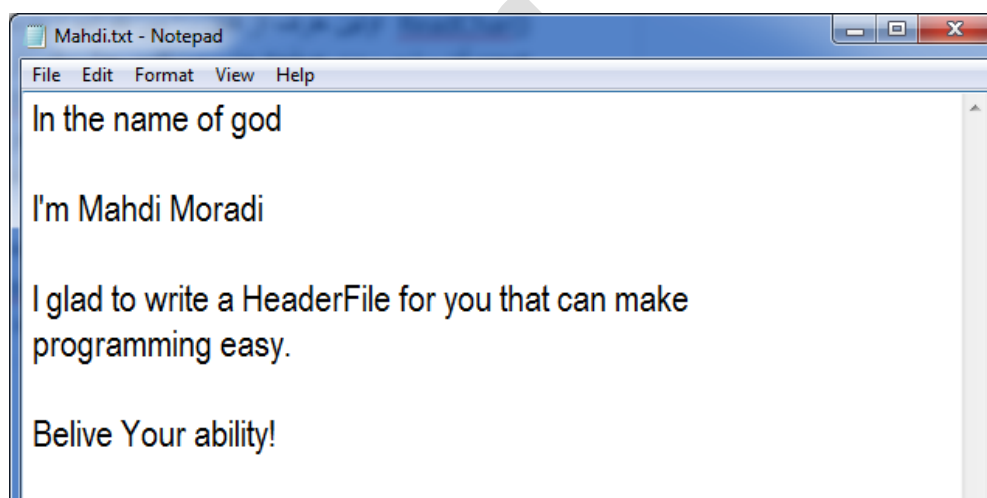
```
Char ch ;  
ch = File.ReadChar();  
cout << ch;  
ch = File.ReadChar();  
cout << ch;
```

خروجی: In:

در برنامه بالا یک متغیر از نوع کاراکتر تعریف کرده ایم و با تابع ReadChar() اولین حرف از فایل را می خوانیم و با دستور خروجی cout آن را بر روی صفحه چاپ می کنیم. حال یک بار دیگر ReadChar() را فراخوانی می کنیم و مقدار آن را در ch قرار می دهیم در این صورت مقدار کنونی متغیر ch برابر با دومین حرف از فایل ما خواهد بود و سپس حرف دوم فایل را چاپ می کنیم می توانیم آن قدر این روند را تکرار کنیم تا در نهایت به انتهای فایل برسیم در حالت پیشفرض اگر ReadChar() رسیدن به انتهای فایل را تشخیص بدهد کاراکتر "*" را

را برگشت می دهد همچنین می توانید از تابع EndFile() برای بررسی رسیدن به انتهای فایل استفاده کنید به مثال پایین دقت کنید:

فایل متنی Mahdi.txt در مسیر C:\Mahdi.txt



متن برنامه:

```
1  #include "iostream"
2  #include "File.h"
3
4  using namespace std;
5
6  int main(){
7
8      File.SetPath("C:\\Mahdi.txt");
9
10     char ch = File.ReadChar();
11
12     do{
13
14         cout << ch ;
15
16     }while( (ch = File.ReadChar()) != File.EndFile() );
17
18     return 0;
19 }
```

خروجی:

```
C:\Users\MahdiCo\Desktop\a.exe
In the name of god
I'm Mahdi Moradi
I glad to write a HeaderFile for you that can make
programming easy.
Belive Your ability!

-----
Process exited after 0.06785 seconds with return value 0
Press any key to continue . . .
```

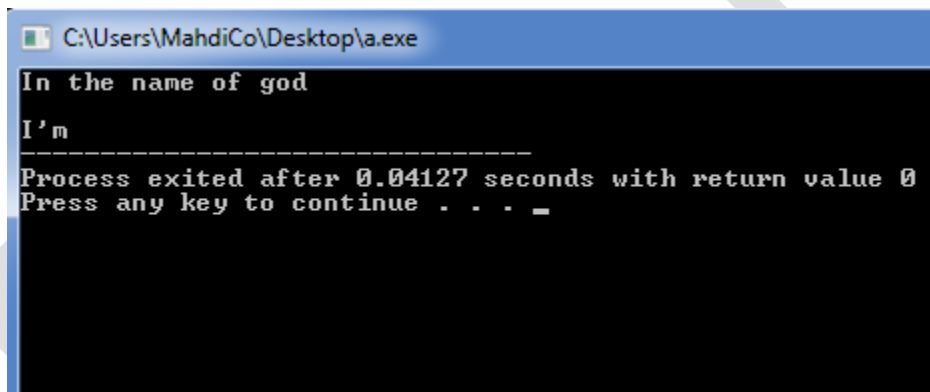
شما می توانید خودتان مقداری را برای کاراکتر انتهای فایل در نظر بگیرید که این کار به کمک تابع `SetEnd()` انجام می شود فقط درباره تنظیم انتهای فایل باید حواستان به یک نکته بسیار مهم باشد و آن هم اینکه اگر انتهای فایل خود را با یک کاراکتر موجود در متن اصلی برابر قرار دهید در خواندن فایل اشکالاتی ایجاد می شود برای درک این موضوع به مثال پایین دقت کنید:

فایل متنی همان فایل سابق می باشد یعنی `Mahdi.txt`

متن برنامه اصلی:

```
1  #include "iostream"
2  #include "File.h"
3
4  using namespace std;
5
6  int main(){
7
8      File.SetPath("C:\\Mahdi.txt");
9      File.SetEnd('M');
10     char ch = File.ReadChar();
11
12     do{
13
14         cout << ch ;
15
16     }while( (ch = File.ReadChar()) != File.EndFile() );
17
18     return 0;
19 }
```

خروجی:



```
C:\Users\MahdiCo\Desktop\a.exe
In the name of god
I'm
-----
Process exited after 0.04127 seconds with return value 0
Press any key to continue . . . _
```

همانطور که در بالا نیز مشاهده می کنید تا قابل از کلمه Mahdi که حرف M دارد از فایل خوانده شد و این به این دلیل است که ما انتهای فایل را حرف M در نظر گرفته بودیم و برنامه با تصور رسیدن به انتهای فایل خواندن را به پایان می رساند، پس می توان نتیجه گرفت که باید در تنظیم کاراکتر انتهایی بسیار دقت کرد!

تابع ReadWord()

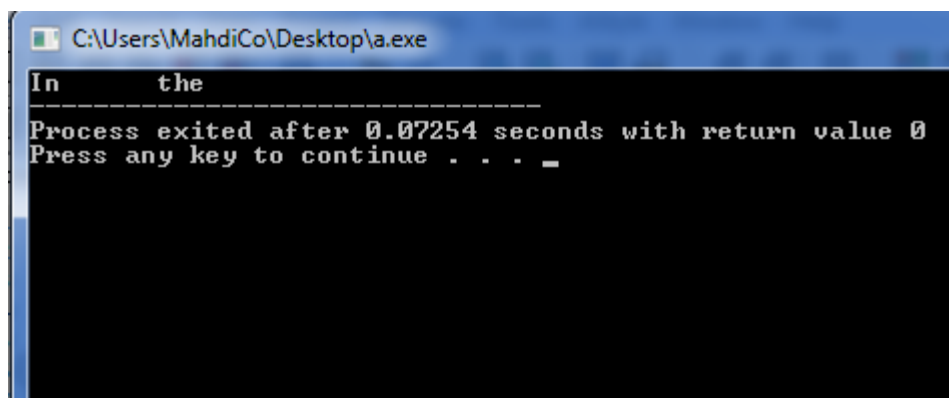
این تابع دقیقا عملکردی مشابه با ReadChar() دارد البته با این تفاوت که بجای دریافت یک کاراکتر از متن یک کلمه را می خواند برای بهتر فهمیدن این موضوع به مثال زیر دقت نمایید:

فایل متنی همان فایل Mahdi.txt در مسیر C:\Mahdi.txt می باشد.

متن برنامه:

```
1  #include "iostream"
2  #include "File.h"
3
4  using namespace std;
5
6  int main()
7  {
8      File.SetPath("C:\\Mahdi.txt");
9
10     string str = File.ReadWord();
11
12     cout << str;
13
14     str = File.ReadWord();
15
16     cout << "\t" << str;
17
18
19     return 0;
20 }
```

خروجی:



```
C:\Users\MahdiCo\Desktop\1.exe
In the
-----
Process exited after 0.07254 seconds with return value 0
Press any key to continue . . . _
```

در بالا مشاهده می کنید دو کلمه اول خوانده شد همانطور که ملاحظه می کنید منظور از کلمه در اینجا خواندن کاراکتر ها تا اولین کاراکتر غیر فضای خالی می باشد.

نکته مهم دیگر اینکه در مقدار برگشتی تابع `ReadWord()` بر خلاف `ReadChar()` که از نوع کاراکتری است از نوع `String` یا رشته ای می باشد.

به عنوان آخرین تابع هم `GetPath()` را به شما معرفی می کنیم که آدرس تنظیم شده توسط `SetPath()` را برای ما باز می گرداند.

مثال :

```
File.SetPath("C:\Mahdi.txt");
```

```
Cout << File.GetPath();
```

به وضوح خروجی این برنامه همان عبارت "C:\Mahdi.txt" می باشد.

بسیار ممنون و متشکر که تا انتهای این فایل ما را همراهی کرده اید
امیدواریم که آن هدر فایل به همراه این راهنمای ارائه شده مورد توجه
شما عزیزان قرار گرفته باشد.

در صورت وجود هر گونه اشکال در فایل خوشحال می شویم آن را به
ما گزارش دهید. همچنین دقت کنید که با باز نمودن هدر فایل مذکور شما
قادر به ایجاد تغییر در کد های موجود هستید و این فایل کاملاً مطابق با
سیاست های متن باز است.
ارتباط با ما:



mahdymorady521@gmail.com



@mahdi_moradi19

پایان