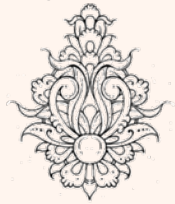
مبانی برنامه‌نویسی
(۱۳۹۹-۱۳-۱۱)
جلسه‌ی یازدهم

عبارت‌های ریاضی
روندنما

دانشگاه شهید بهشتی
پاییز ۱۳۹۲
دانشکده‌ی مهندسی برق و کامپیوتر
احمد محمودی ازناوه

فهرست مطالب

- مروری بر جلسه‌ی پیش
- اولویت‌ها
- عملگرهای یکانی
- الگوریتم
- روندنما(فلوچارت)



مثال

```c
#include<stdio.h>
int main(){
    int num=5;
    float num=5.1;
    printf("num=%d\tnum=%f",num,num);
    return(0);
}//end main
```

```
ahmad@ubuntu:~/MyData/courses/ITP$ gcc example14.c
example14.c: In function ‘main’:
example14.c:4: error: conflicting types for ‘num’
example14.c:3: note: previous definition of ‘num’ was here
example14.c:5: warning: format ‘%d’ expects type ‘int’, but argument 2 has type
‘double’
```

نمی‌توانیم دو متغیر هم‌نام تعریف کنیم!

شیوه‌ی جدید و قدیمی

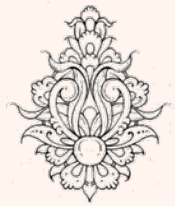
```
/*
   An old-style C++ program.
*/

#include <iostream.h>

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

```
/*
   A modern-style C++ program that uses
   the new-style headers and a namespace.
*/
#include <iostream>
using namespace std;

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

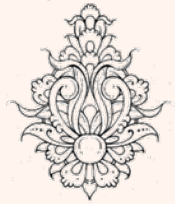
شیوه‌ی جدید و قدیمی (ادامه...)

- سرآیندهایی که با پسوند h. مشخص می‌گردند، حتماً نشان‌گر یک فایل خواهند بود. (شیوه‌ی قدیمی استفاده از سرآیندها)

```
Stdio.h   (file name)
```

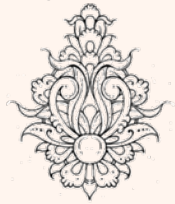
- در شیوه‌ی نوین نام سرآیند لزوماً نشان‌دهنده‌ی یک فایل نبوده، می‌تواند نگاشتی از یک فایل به‌شمار رود.

```
<iostream>
```



شیوه‌ی جدید و قدیمی (ادامه...)

- هنگامی‌که یک سرآیند include می‌گردد، محتویات آن در محدوده‌ای واقع شده است که به آن namespace گفته می‌شود.
- مولفه‌هایی که در یک namespace وجود دارند، مستقل از مولفه‌هایی خواهند بود که در namespace دیگر است.
- C++ Standard Library در namespace با نام std تعریف شده است.
- هر آن‌چیز که در namespace با نام std است می‌باید با پیشوند std آورده شود.



namespace

foo.h

```
// This DoSomething() adds it's parameters
int DoSomething(int nX, int nY)
{
    return nX + nY;
}
```

goo.h

```
// This DoSomething() subtracts it's parameters
int DoSomething(int nX, int nY)
{
    return nX - nY;
}
```

namespace.cpp

```
#include "foo.h"
#include "goo.h"
#include <iostream>

int main()
{
    using namespace std;
    cout << DoSomething(4, 3); // which DoSomething will we get?
    return 0;
}
```

```
ahmad@ubuntu:~/Courses/ITP$ g++ -Wall namespace.cpp foo.h goo.h -o test
In file included from namespace.cpp:2:
goo.h: In function ‘int DoSomething(int, int)’:
goo.h:2: error: redefinition of ‘int DoSomething(int, int)’
foo.h:2: error: ‘int DoSomething(int, int)’ previously defined here
```

Namespace (ادامه...)

foo.h

```
namespace foo{
        // This DoSomething() adds it's paramet
        int DoSomething(int nX, int nY)
        {
            return nX + nY;
        }
}
```

goo.h

```
namespace goo{
        // This DoSomething() subtracts it's parameters
        int DoSomething(int nX, int nY)
        {
            return nX - nY;
        }
}
```

namespace.cpp

```
#include
#include
#include <iostream>

int main()
{
    using namespace std;
    using namespace foo;
   cout <<  DoSomething(4, 3);
   cout << '\n';

   cout << goo::DoSomething(4, 3);
   cout << '\n';
   return 0;
}
```

```
71ahmad@ubuntu:~/Courses/ITPg++ -Wall namespace.cpp foo.h goo.h -o test
ahmad@ubuntu:~/Courses/ITP$ ./test
7
1
```

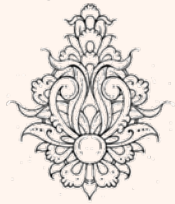
C, C++

- برای این که بتوانیم هنگام اجرای برنامه مقادیری را وارد کنیم از عملگر ورودی >> استفاده می‌کنیم.

```
cin >> variable;
```

- برای این که بتوانیم هنگام اجرای برنامه مقادیری را چاپ کنیم.

```
cout << variable;
```

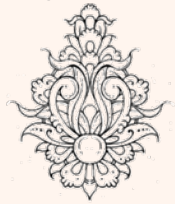


اولویت‌ها

- تقدم عملیات ریاضی به صورت زیر است:

عملگر	ترتیب
پرانتز ها	()
عملگر های ضرب و تقسیم و باقی مانده	* / %
عملگرهای جمع و تفریق	+ -

- در صورت وجود بیش از یک نمونه در یک عبارت به ترتیب از چپ به راست ارزیابی می‌شوند.



اپراتورهای افزایش و کاهش

- عملگر ++:
  - پیشوندی

```
++P
```

    - یک واحد افزایش یافته سپس فرآیند صورت می‌گیرد
  - پسوندی

```
P++
```

    - فرآیند صورت می‌گیرد، سپس یک واحد افزایش می‌یابد
- عملگر --:
  - پیشوندی

```
--P
```

    - یک واحد کاهش یافته سپس فرآیند صورت می‌گیرد
  - پسوندی

```
P--
```

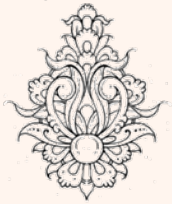
    - فرآیند صورت می‌گیرد، سپس یک واحد کاهش می‌یابد

این عملگرها بالاترین اولویت را دارند

مثال

```c
#include <stdio.h>
int main(){ // shows the difference between m++ and ++m:
    int m,n;
    m = 88;
    n = ++m; // the pre-increment operator is applied to m
    printf("m=%d, n=%d\n",m, n);
    m = 88;
    n = m++; // the post-increment operator is applied to m
    printf("m=%d, n=%d\n",m, n);
    return 0;
}// end of main
```

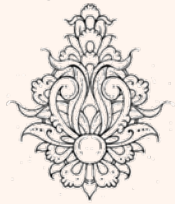
```
m=89, n=89
m=89, n=88
```



مثال

```
int main()
{ // shows the difference between m++ and ++m:
int m,n;
m = 88;
n = ++m; // the pre-increment operator is applied to m
cout << "m = " << m << ", n = " << n << endl;
m = 88;
n = m++; // the post-increment operator is applied to m
cout << "m = " << m << ", n = " << n << endl;
}
```

```
m = 89, n = 89
m = 89, n = 88
```

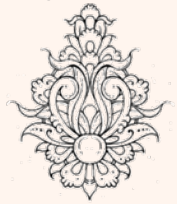


الگوریتم‌ها

- هر مسأله‌ی محاسباتی را می‌توان با اجرای دنباله‌ای از اعمال با ترتیبی مشخص حل نمود.
- رویه‌ی حل یک مساله را بر حسب
  - اعمالی که باید اجرا شوند
  - ترتیب اجرای اعمال

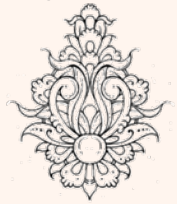
الگوریتم می‌نامند.

- الگوریتم مجموعه‌ای متناهی از دستورالعمل‌هاست، که به ترتیب خاصی اجرا می شوند و مسأله‌ای را حل می‌کنند. به عبارت دیگر یک الگوریتم روشی گام به گام برای حل مسئله است.



حل مساله

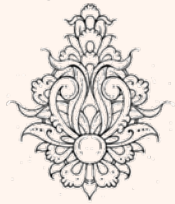
- درک مساله
- نوشتن الگوریتم
  - متغیرها
    - ورودی
    - میانی
    - خروجی
- بسط الگوریتم
- آزمایش الگوریتم



مثال

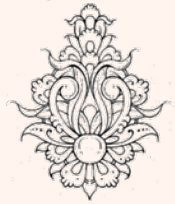
- سه عدد داریم که می‌خواهیم میانگین و مجموعشان را حساب کنیم به چه متغیرهایی نیاز است؟

ورودی
A,B,C
میانی
خروجی Sum و Ave



دستورالعمل‌ها در الگوریتم

- **الگوریتم از مجموعه‌ای دستورالعمل‌ها تشکیل شده است:**
  - ورودی
    - بگیر-دریافت کن
  - خروجی
    - نمایش بده- چاپ کن
  - محاسباتی
    - محاسبات و سپس انتساب



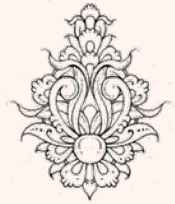
دستورالعمل‌ها در الگوریتم (ادامه...)

- شرطی
  - روند اجرای الگوریتم به واسطه‌ی اتخاذ تصمیمی درست برگزیده می‌شود.

If شرط (ها) **then** دستور (ات)

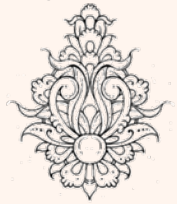
If شرط (ها) **then** دستور (ات) **else** دستور (ات)

  - در بررسی شرط‌ها معمولاً از عملگرهای مقایسه‌ای استفاده می‌شود.
- تکرار (حلقه‌ها)
  - تا زمانی که شرط برقرار است فرآیندی به صورت متوالی اجرا می‌گردد.



شبه‌کد

- شبه‌کد یک زبان ساختگی و غیر رسمی است که به برنامه‌نویس در توسعه‌ی الگوریتم کمک می‌نماید.
- شبه کد زبان برنامه‌نویسی نیست ولی کار با آن ساده است.
- شبه‌کدی که خوب نوشته شود را، به آسانی می‌توان با یک زبان برنامه‌نویسی پیاده‌سازی نمود.
- شبه‌کد می‌باید به اندازه‌ی کافی با جزییات نوشته شود تا پیاده‌سازی آن به زبان برنامه‌نویسی ساده گردد.



مثال

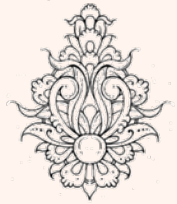
- برنامه‌ای بنویسید که دو عدد از ورودی گرفته حاصل‌جمع را محاسبه و نمایش دهد:

متغیرها

ورودی
A,B
میانی

خروجی
Sum

1. شروع
2. عدد A را وارد کن
3. عدد B را وارد کن
4. دو عدد را با هم جمع کن و در Sum بریز
5. Sum را چاپ کن
6. پایان



مثال

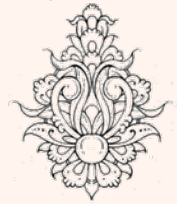
- برنامه‌ای بنویسید که سه عدد را گرفته میانگین مقادیر را چاپ کند.

متغیرها

ورودی
A,B و C
میانی

خروجی
Ave

1. شروع
2. عدد A را دریافت کن
3. عدد B را دریافت کن
4. عدد C را دریافت کن
5. Ave ⟸ (A+B+C)/3
6. Ave را چاپ کن
7. پایان

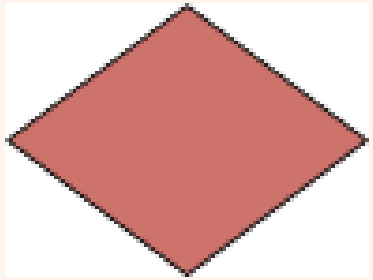


فلوچارت (روندنما)

- روندنما مانند شبه‌کد برای توسعه و نمایش الگوریتم مفید است.

نماد عمل (محاسبات)

نماد شروع - پایان



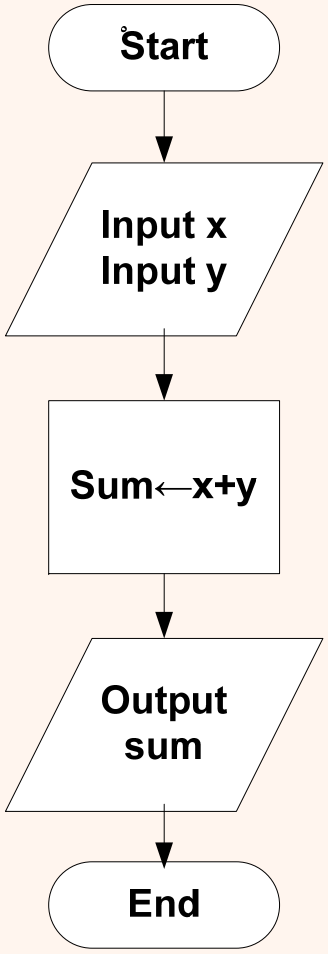
نماد تصمیم‌گیری (شرط‌ها)

ورودی - خروجی

مثال

- برنامه‌ای بنویسید که دو عدد از ورودی گرفته حاصل‌جمع را محاسبه و نمایش دهد:

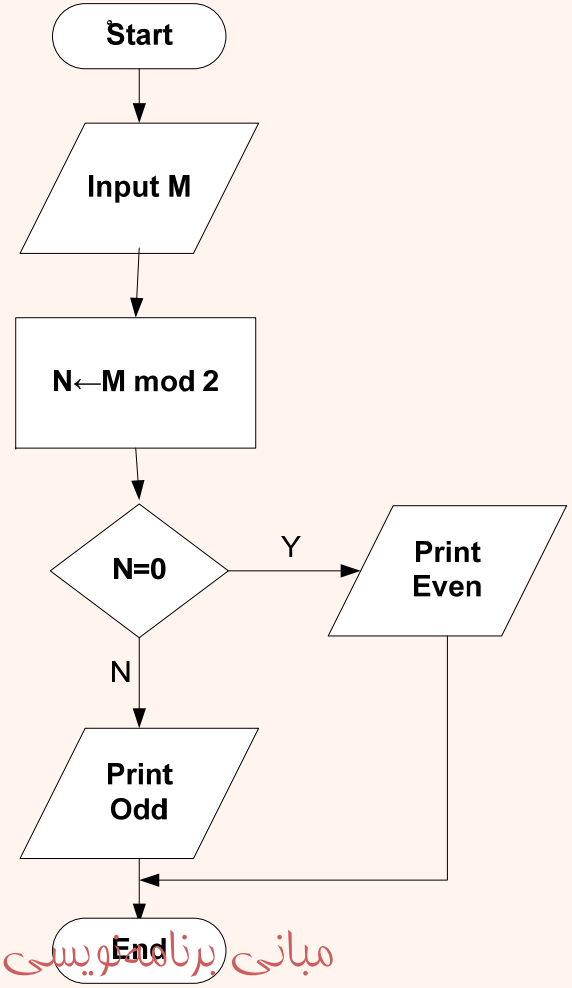
```
int main()
     {   // reads two integer from input:
         int x,y,Sum;
         cout << "Enter first number: ";
         cin >> x;
         cout << "Enter Second number: ";
         cin >> y;
         Sum=x+y;
         cout << "Sumation  is: " << Sum << endl;
         return 0;
     }
```




```
Enter first number: 2
Enter Second number: 3
Sumation  is: 5
```

مثال

- الگوریتم و روندنمایی بنویسید که عدد طبیعی و دلخواه M را دریافت کرده، زوج یا فرد بودن آن‌را معین نماید.



1. شروع
2. M را دریافت کن
3. N ⇐ M mode 2
4. اگر N=0 است نمایش بده عدد زوج است در غیر این‌صورت عدد فرد است را چاپ کن.
5. پایان