اصول کامپیوتر ۱

مبانی کامپیوتر و برنامه‌سازی

«جلسه‌ی ششم»

دانشکده‌ی علوم ریاضی - دانشگاه شهید بهشتی

نیم‌سال اول ۹۰-۱۳۸۹

مدرس: سید علی کتان‌فروش

زبان برنامه‌نویسی C

- زبان برنامه‌نویسی C در سال ۱۹۷۲ توسط Dennis Ritchie در آزمایشگاه‌های شرکت Bell ابداع شد.
- اولین کتاب آموزش زبان C توسط Kernighan و Ritchie در ۱۹۷۸ منتشر شد.
- استانداردسازی‌های متعددی بر روی این زبان صورت گرفته است که آخرین آن‌ها ISO/IEC 1999 است که به C99 شهرت دارد.

زبان برنامه‌نویسی C

- این زبان برنامه‌نویسی به طور خاص به منظور برنامه‌نویسی سیستم‌عامل Unix ابداع شد.
- در مقایسه با دیگر زبان‌های برنامه‌نویسی، دستورات زبان C از نظر معنائی اختلاف کمی با دستورات زبان ماشین دارند و از این رو، C زبانی سطح میانی به حساب می‌آید.

زبان برنامه‌نویسی C

- زبان برنامه‌نویسی C، زبانی پروسه‌ای (procedural) و ساخت‌یافته (structured) است.
- زبان C قابل انعطاف و قدرتمند است یعنی محدودیت بسیاری کمی برای برنامه نویس برای استفاده از امکانات سخت‌افزار و سیستم عامل ایجاد می‌کند.

زبان برنامه‌نویسی C

- زبان C، زبان قابل حملی است.

  یعنی برنامه‌ی نوشته‌شده به زبان C را به سادگی می‌توان بر روی هر سخت‌افزار و هر سیستم‌عاملی، کامپایل و اجرا کرد.

- هرچند قابل حمل بودن زبان C در آن زمان، ویژگی منحصربفردی بوده است اما امروزه زبان‌های برنامه‌نویسی متعددی وجود دارند که چنین ویژگی‌ای دارند.

الگوی کلی یک برنامه‌ی ساده در C

```
#include <stdio.h>
#define MAX_SIZE 20

int main( int argc, char *argv[]){
  char s[MAX_SIZE];
  scanf("%s",s);
  printf("Hello %s\n", s);
  return 0;
}
```

الگوی کلی یک برنامه‌ی ساده در C

دستورات pre-compiler

```
#include <stdio.h>
#define MAX_SIZE 20
```

```
int main( int argc, char *argv[]){
  char s[MAX_SIZE];
  scanf("%s",s);
  printf("Hello %s\n", s);
  return 0;
}
```

تابع main

دستورات برنامه در اینجا نوشته می‌شوند

دستورات pre-compiler

- هر دستور pre-compiler در یک خط نوشته می‌شود و با یک علامت # شروع می‌شود.
- اصولاً این دستورات ارتباطی به الگوریتم و منطق پیاده‌سازی شده در برنامه ندارند و از آن‌ها تنها به منظور تنظیم نحوه‌ی کامپایل برنامه و دادن دستور به کامپایلر استفاده می‌شود.

تابع main

- مجموعه دستورات یک برنامه در زبان برنامه‌نویسی C در بخش‌هایی که اصطلاحاً آن‌ها را توابع می‌نامیم نوشته می‌شوند.
- هر تابع دارای یک header (شامل نام تابع و ...) و یک بدنه (body) است.
- برنامه برای اینکه اجرا شود باید تابعی به نام main داشته باشد.

تابع main

- بدنه‌ی تابع در یک بلوک (block) قرار می‌گیرد.
- اصطلاحاً، مجموعه دستورالعملهایی که بین { و } قرار می‌گیرند را یک بلوک می‌گویند.
- header برای تابع mainباید به صورتی که در الگوی فوق آمده است نوشته شود.

دستورات در زبان C

- زبان C به شکل حروف از حیث کوچک یا بزرگ بودن حساس است. اصطلاحاً case-sensitive است.
- مثلاً اگر دستورالعملی در زبان C به طور قراردادی با حروف کوچک تعریف شده است استفاده از آن با حروف بزرگ در برنامه باعث بروز خطا هنگام کامپایل می‌شود.
- همچنین اگر متغیری را با نامی با حروف کوچک تعریف کرده‌اید نمی‌توانید در برنامه آنرا با حروف بزرگ مورد استفاده قرار دهید.

دستورات در زبان C

- در عوض، حساس بودن زبان C به شکل حروف به برنامه‌نویس کمک می‌کند تا حوزه‌ی وسیع‌تری از کلمات را برای نامگذاری متغیرهای خود داشته باشد.
- مثلاً در یک برنامه می‌توانید در کنار هم، متغیرهایی با نام serverTime, ServerTime, servertime,SERVERTIME, SeRvErTiMe و ... داشته باشیم.

دستورات در زبان C

- هر دستور در یک برنامه‌ی نوشته شده به زبان C با یک علامت ; پایان می‌یابد.
  به این علامت semicolon یا نقطه‌ویرگول می‌گوییم.
- بنابراین، می‌توانید چند دستور را در یک خط بنویسید.
  همچنین می‌توانید یک دستور بلند را در چند خط بنویسید.
  در هر صورت علامت نشان‌دهنده‌ی پایان دستور ; است.

دستورات در زبان C

- در زبان C، اهمیتی ندارد اگر یک یا بیشتر فاصله‌ی خالی (space) بین اجزاء یک دستور قرار گیرد. به عنوان مثال هر سه دستور زیر برای کامپایلر از نظر نگارشی یکسان‌اند.

```
x=y+z;
x = y + z;
   x = y+z;
```

دستورات در زبان C

- برای خوانائی بهتر برنامه، متعارف آن است که دستورات داخل هر بلوک به اندازه‌ی چند space با لبه‌ی سمت چپ بلوک فاصله داشته باشند.
- این عمل را اصطلاحاً indentation (دندانه‌گذاری یا کنگره‌بندی) برنامه می‌گوئیم.

الگوی کلی یک برنامه‌ی ساده در C

دستورات pre-compiler

```
#include <stdio.h>
#define MAX_SIZE 20

int main( int argc, char *argv[]){
  char s[MAX_SIZE];
  scanf("%s",s);
  printf("Hello %s\n", s);
  return 0;
}
```

تابع main

دستورات برنامه در اینجا نوشته می‌شوند

اولین دستور.

اعلان متغیر Variable declaration

- در برنامه‌ای که به زبان C نوشته می‌شود تمام متغیرها می‌باید پیش از استفاده، اعلان (declare) شوند.
- منظور از اعلان یک متغیر، تعیین «نوعِ داده‌ای» (data-type) متغیر است.

به عبارت ساده، برنامه‌نویس باید تصریح کند که متغیر مورد استفاده‌اش از کدامیک از انواع عدد صحیح، عدد اعشاری، حرفی و ... است.

اولین دستور.

اعلان متغیر Variable declaration

- اولین دستور در برنامه‌ی فوق، تعریف متغیری از نوع حرفی (char) با نام s است که می‌تواند کلماتی با حداکثر ۲۰ حرف را در خود نگه دارد:

```
char s[MAX_SIZE];
```

- توجه کنید که اعلان متغیر به معنای مقداردهی اولیه‌ی متغیر نیست و مقداردهی متغیر وظیفه‌ی ضروری دیگری است که برنامه‌نویس باید مورد توجه قرار دهد.

دستور دوم.

خواندن اطلاعات از ورودی

- دومین دستور، مقدار متغیرs را از کاربر می‌خواند:

```
scanf("%s", s);
```

چاپ خروجی

- سومین دستور، رشته‌ی حرفی داده شده به متغیرs را مقابل Hello چاپ می‌کند:

```
printf("Hello %s\n", s);
```

دستور چهارم.

پایان برنامه

- با آخرین دستور، اجرای برنامه پایان می‌یابد:

```
return 0;
```

دستورات کلیدی زبان C

- همانطور که اشاره شد زبان C بسیار نزدیک به زبان ماشین است از همین رو، مجموعه دستورات اصلی زبان C که آن‌ها را کلمات کلیدی (keywords) می‌نامیم مجموعه‌ی بسیار کوچکی است.

دستورات کلیدی زبان C

Only 32 keywords!

```
auto   break  case   char   const
continue default  do double
else   enum   extern   float  for
goto  if  int  long   register
return   short  signed   sizeof
static struct  switch    typedef
union  unsigned void   volatile
while
```

All in lower case

کتابخانه‌ی توابع Function library

- تنها ۳۲ دستورالعمل اصلی در زبان C وجود دارد.
- بدیهی است دستورات متعدد دیگری نیز برای برنامه‌نویسی لازم‌اند که در این مجموعه قرار ندارند.

کتابخانه‌ی توابع Function library

- ایده‌ی طراحان زبان C بر این است که مجموعه‌ی کوچکی از دستورات اساسی برنامه‌نویسی به عنوان کلمات کلیدی زبان تعریف شوند و دیگر دستورات مورد نیاز برای برنامه‌نویسی در قالب مجموعه‌هایی که آن‌ها را اصطلاحاً کتابخانه‌ی توابع می‌گوئیم در کنار زبان C توسعه داده شوند.

کتابخانه‌ی توابع Function library

- به طور نمونه، برای انجام عملیات ورودی/خروجی هیچ دستوری در مجموعه‌ی کلمات کلیدی زبان C تعریف نشده است.
- در واقع، دو دستور scanf و printf در برنامه‌ی فوق، توابعی از کتابخانه‌ی stdio هستند و دستور

```
#include <stdio.h>
```

به کامپایلر اعلام می‌کند که برنامه می‌خواهد از برخی از توابع این کتابخانه استفاده کند.

کتابخانه‌ی توابع Function library

- کتابخانه‌های متعددی به منظور توسعه‌ی زبان C در کاربردهای مختلف نوشته شده‌اند.
- بسیاری از این کتابخانه‌ها به طور تخصصی برای کاربردهایی خاص نوشته شده‌اند و ممکن است به طور رایگان یا تجاری در اختیار برنامه‌نویسان علاقه‌مند قرار گیرند.

کتابخانه‌ی توابع Function library

- تعدادی از کتابخانه‌ها به عنوان کتابخانه‌های استاندارد معرفی شده‌اند که هر کامپایلر استانداردی می‌بایست امکان استفاده از آن‌ها را برای برنامه‌نویس فراهم کند. مانند
  - math
  - stdlib (cstdlib)
  - stdio (cstdio)
  - iostream
  - ...

زبان برنامه‌نویسی C++

- زبان برنامه‌نویسی C++ در سال ۱۹۸۳ توسط Bjarne Stroustrup در قالب توسعه‌ای بر زبان C معرفی شد.
- هدف از توسعه‌ی زبان C به C++، به طور عمده افزودن قابلیت‌هایی برای برنامه‌نویسی شیءگرا (object-oriented) بود.
- C++ همانند Cیک زبان همه‌منظوره است
- علاوه بر ویژگی‌های مناسب برای برنامه‌نویسی شیءگرا، C++ امکان برنامه‌نویسی ژنریک (generic programming) را نیز فراهم می‌کند.
- آخرین استانداردسازی C++ در سال 2003 توسط ISO/IEC صورت گرفته است.

الگوی کلی یک برنامه‌ی ساده در C++

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
  string s;
  cin >> s;
  cout << "Hello " << s << endl;
  return EXIT_SUCCESS;
}
```

الگوی کلی یک برنامه‌ی ساده در C++

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
  string s;
  cin >> s;
  cout << "Hello " << s << endl;
  return EXIT_SUCCESS;
}
```

دستورات pre-compiler

تابع main

دستورات برنامه در اینجا نوشته می‌شوند

الگوی کلی یک برنامه‌ی ساده در C++

کتابخانه‌ها بدون پسوند .h استفاده می‌شوند.

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
  string s;
  cin >> s;
  cout << "Hello " << s << endl;
  return EXIT_SUCCESS;
}
```

این دستور برای استفاده‌ی آسانتر از کتابخانه‌ها لازم است.

- توجه کنید که توسعه‌ی زبان C به C++ به معنای اضافه شدن دستوراتی مثل دستورات ورودی/خروجی به زبان C نیست.
- درواقع چنین دستوراتی همچنان در قالب کتابخانه‌ها در اختیار برنامه‌نویس قرار می‌گیرند.
- کلمات کلیدی جدیدی که در C++ اضافه شده‌اند اصولاً دستوراتی مرتبط با مفاهیم برنامه‌نویسی شیءگرا و برنامه‌نویسی ژنریک هستند.

دستورات کلیدی زبان C++

```
C keywords +

bool catch class delete explicit
false inline mutable namespace
new operator private protected
public template this throw true
try typeid typename using
virtual wchar_t
```

کتابخانه‌های C++

- در این درس ما از امکانات فراهم شده در کتابخانه‌هایی که مبتنی بر دیدگاه شیءگرایانه و برنامه‌نویسی ژنریک توسعه داده شده‌اند مثل string، iostream و vector استفاده می‌کنیم اما الگوریتم‌ها را در چارچوب این دیدگاه‌ها پیاده‌سازی نمی‌کنیم.

الگوی کلی یک برنامه‌ی ساده در C++

```
#include <iostream>
#include <string>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
  string s;
  cin >> s;
  cout << "Hello " << s << endl;
  return EXIT_SUCCESS;
}
```

اعلان متغیر

- در ارتباط با حروف، دو نوع داده‌ای در زبان C/C++ وجود دارد:
  - کاراکتر (char) که نوع داده‌ای کاراکترها را شامل می‌شود. مثل 'a' و '2' و '=' و امثال آن‌ها.
  - رشته (string) که شامل دنباله‌ای از هیچ یا یک یا بیشتر کاراکتر می‌شود. مثل "a" ,"abc" و "".
- با آنکه رشته‌های ثابت در زبان C به سادگی مورد استفاده قرار می‌گیرند اما برای تعریف متغیری از نوع رشته، نوع داده‌ای سرراستی وجود ندارد.

اعلان متغیر

- با اولین دستور در برنامه‌ی فوق، متغیری با نام s و از نوع رشته‌ی حرفی تعریف می‌شود:

```
string s;
```

- نوع داده‌ای رشته در C++ در کتابخانه‌ی string تعریف شده است:

```
#include <string>
```

دستور دوم.

دریافت ورودی

- در دومین دستور، مقدار رشته‌ی حرفی s از کاربر گرفته می‌شود:

```
cin >> s;
```

چاپ خروجی

- در سومین دستور، پیام Hello چاپ می‌شود که در مقابل آن محتوای متغیر s قرار داده شده است:

```
cout << "Hello " << s << endl;
```

- cin و cout اشیائی هستند به ترتیب مرتبط با دستگاه ورودی (صفحه کلید) و دستگاه خروجی (صفحه‌ی نمایش) که در کتابخانه‌ی iostream. تعریف شده‌اند

پایان

- آخرین دستور در برنامه‌ی فوق، یعنی

```
return EXIT_SUCCESS;
```

دقیقاً معادل

```
return 0;
```

است که درواقع پایان موفقیت‌آمیز برنامه را به سیستم‌عامل اعلام می‌کند.

پیاده‌سازی نمونه‌هایی از الگوریتم‌های «ورودی-محاسبه-خروجی» در C++

مثال. محاسبه‌ی مقلوب عدد دو رقمی.

N=45 => 54

ورودی. عدد دو رقمی مثبت N.

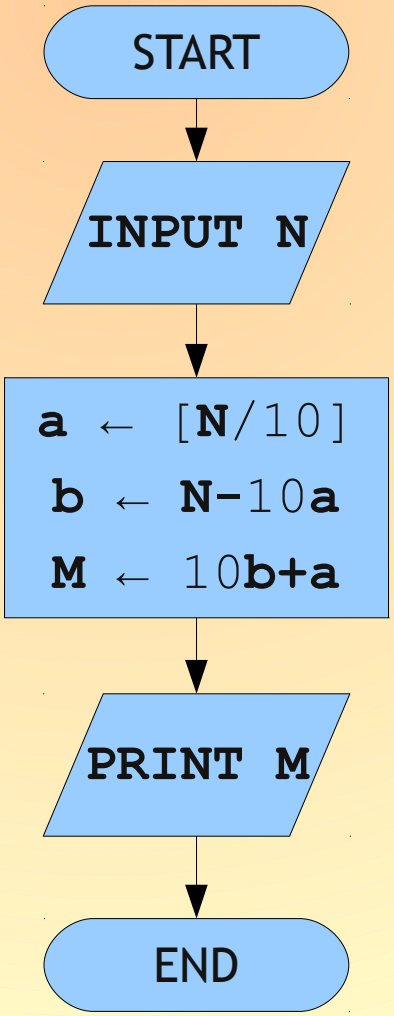
گام ۱. قرار ده $a \leftarrow [N/10]$

گام ۲. قرار ده $b \leftarrow N - 10a$

گام ۳. قرار ده $M \leftarrow 10b + a$

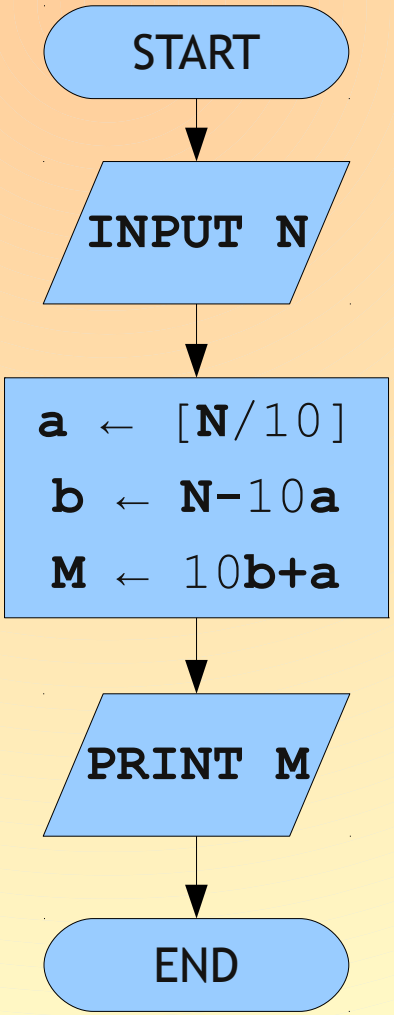
گام ۴. M را به عنوان نتیجه چاپ کن.

گام ۵. پایان.




```
#include <iostream>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
   int N;
   int a;
   int b;
   int M;
   cin >> N;
   a=N/10;
   b=N-10*a;
   M=10*b+a;
   cout << M << endl;
   return EXIT_SUCCESS;
}
```



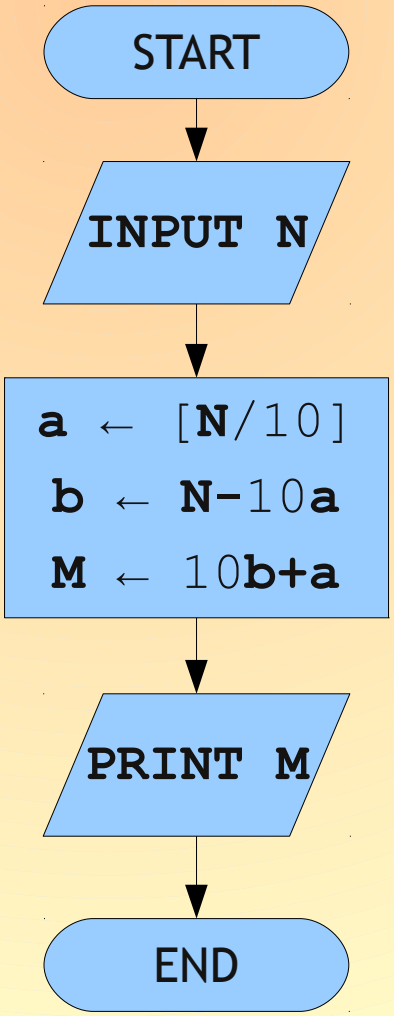

```
#include <iostream>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
   int N;
   int a;
   int b;
   int M;
   cin >> N;
   a=N/10;
   b=N-10*a;
   M=10*b+a;
   cout << M << endl;
   return EXIT_SUCCESS;
}
```

اعلان متغیرها

int نوع داده‌ای اعداد

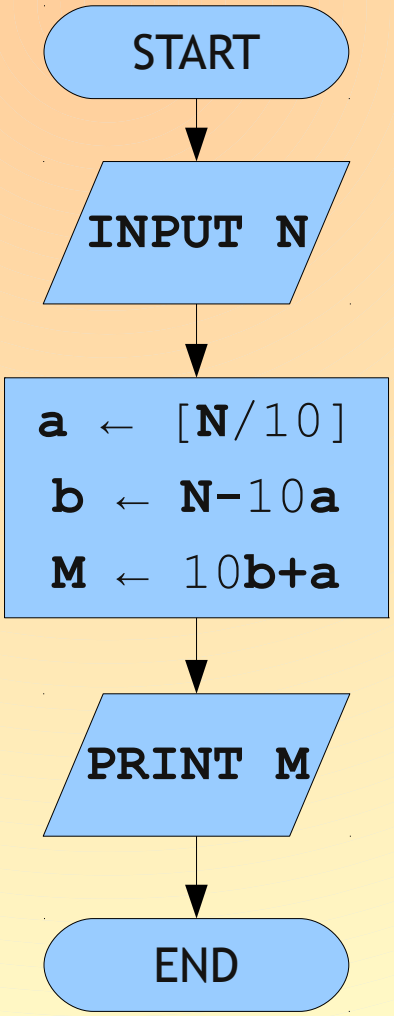
صحیح در زبان C/C++ است




```
#include <iostream>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
  int N,a,b,M;
  cin >> N;
  a=N/10;
  b=N-10*a;
  M=10*b+a;
  cout << M << endl;
  return EXIT_SUCCESS;
}
```

اعلان چند متغیر در یک دستور مجاز است.




```
#include <iostream>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
   int N,a,b,M;
   cin >> N;
   a=N/10;
   b=N-10*a;
   M=10*b+a;
   cout << M << endl;
   return EXIT_SUCCESS;
}
```

در زبان C/C++ وقتی هر دو عملوند تقسیم از نوع عدد صحیح باشند خارج قسمت صحیح (بدون اعشار) محاسبه می‌شود.

```
#include <iostream>
#include <cstdlib>
using namespace std;

int main( int argc, char *argv[]){
  int N,a,b;
  cin >> N;
  a=N/10;
  b=N-10*a;
  cout << b << a << endl;
  return EXIT_SUCCESS;
}
```