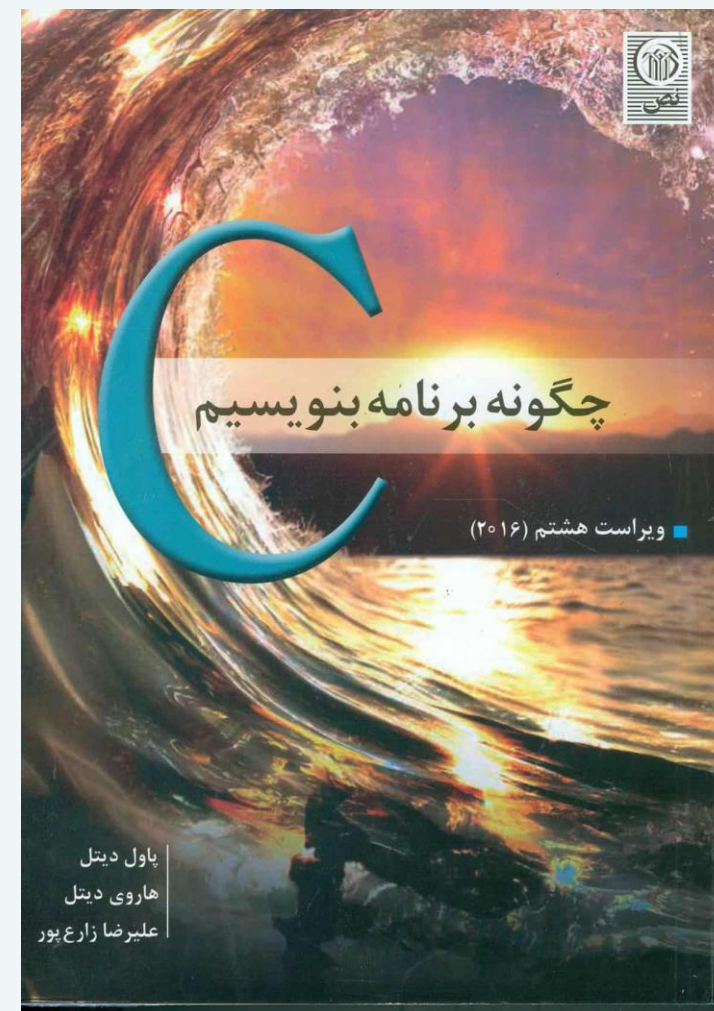
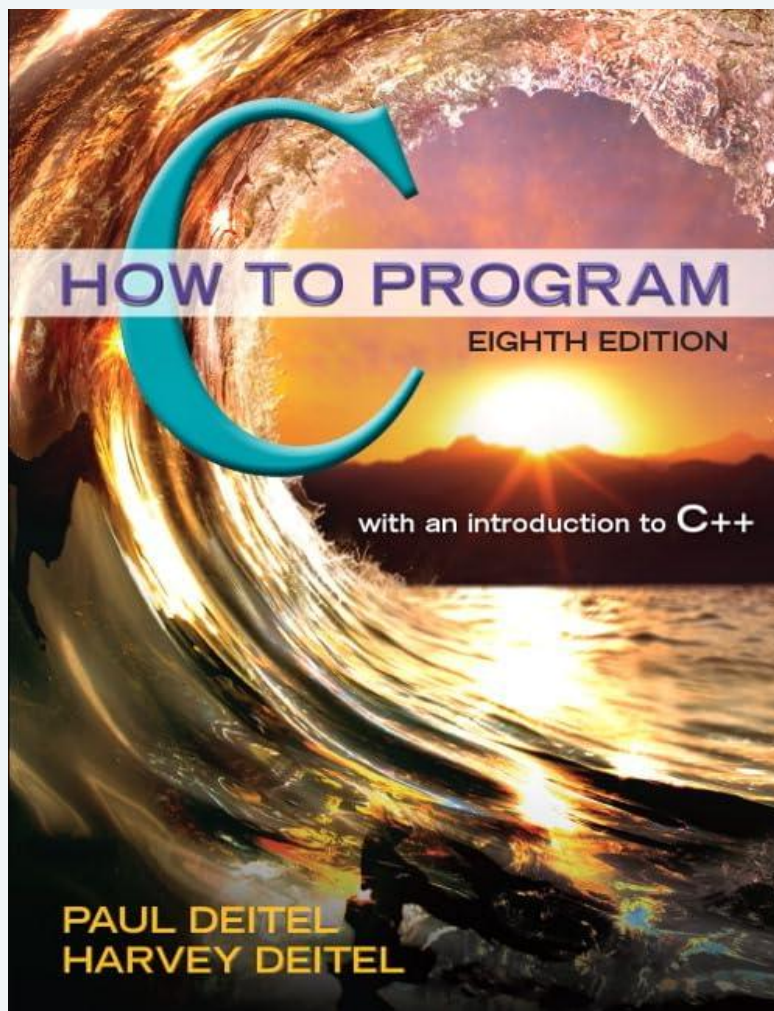


1

INTRO. TO PROGRAMMING



مراجع





کلیات

■ بارم بندی:

۱۲ نمره پایان ترم

۴ میان ترم

۴ پروژه

۲. راه ارتباطی Yasamansabahi@gmail.com

۳. برای دریافت اسلایدها و نمونه کدها :

<https://github.com/Ysabahi/Computer-Programming-C>



آشنایی کلی با برنامه نویسی

آشنایی با برنامه نویسی C

مفاهیم اولیه و پایه، مقدمات برنامه نویسی

فرمت بندی ورودی و خروجی

آشنایی با الگوریتم، فلوچارت و شبه کد

عبارات و دستورالعمل ها

توابع

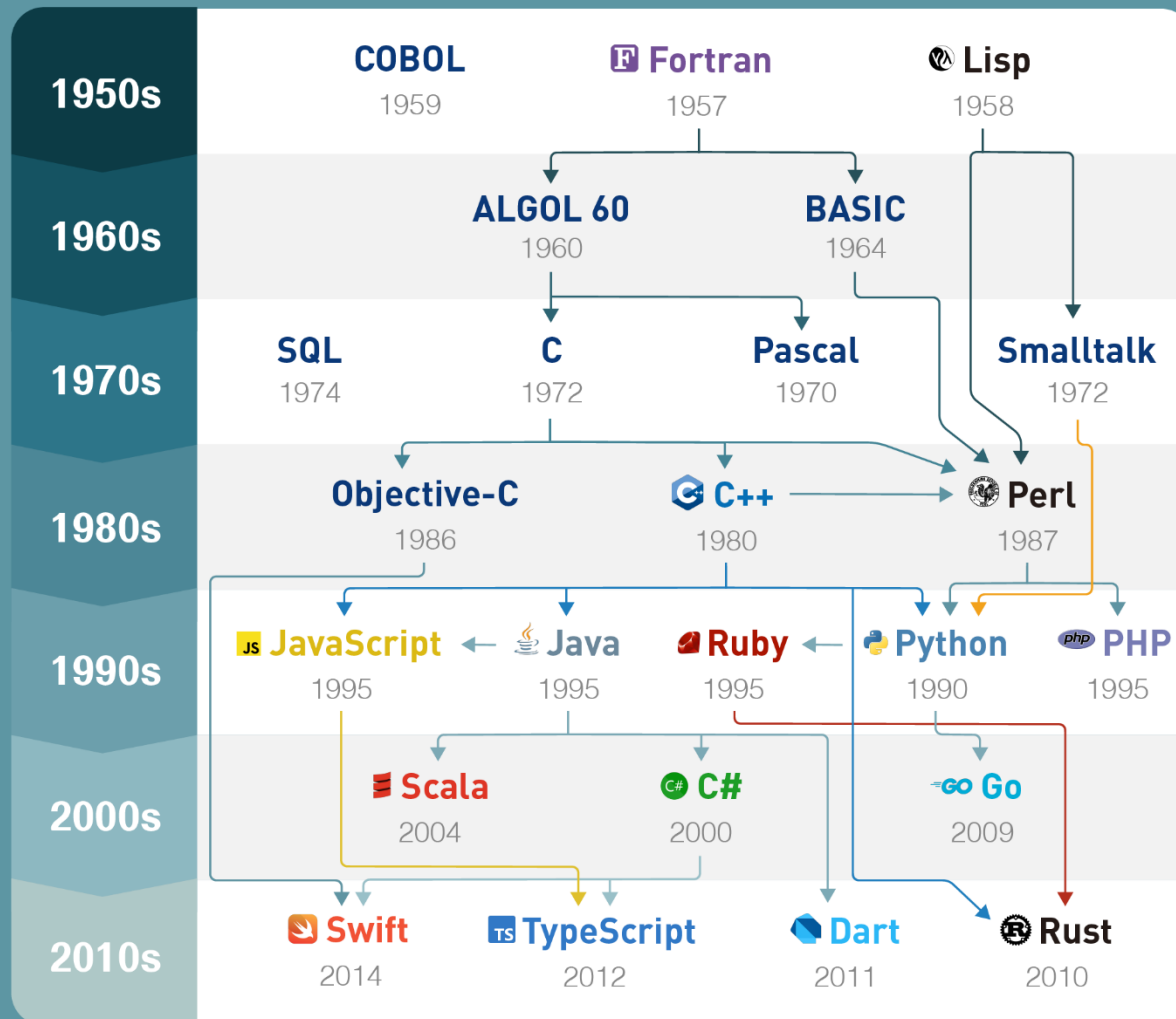
عیب یابی

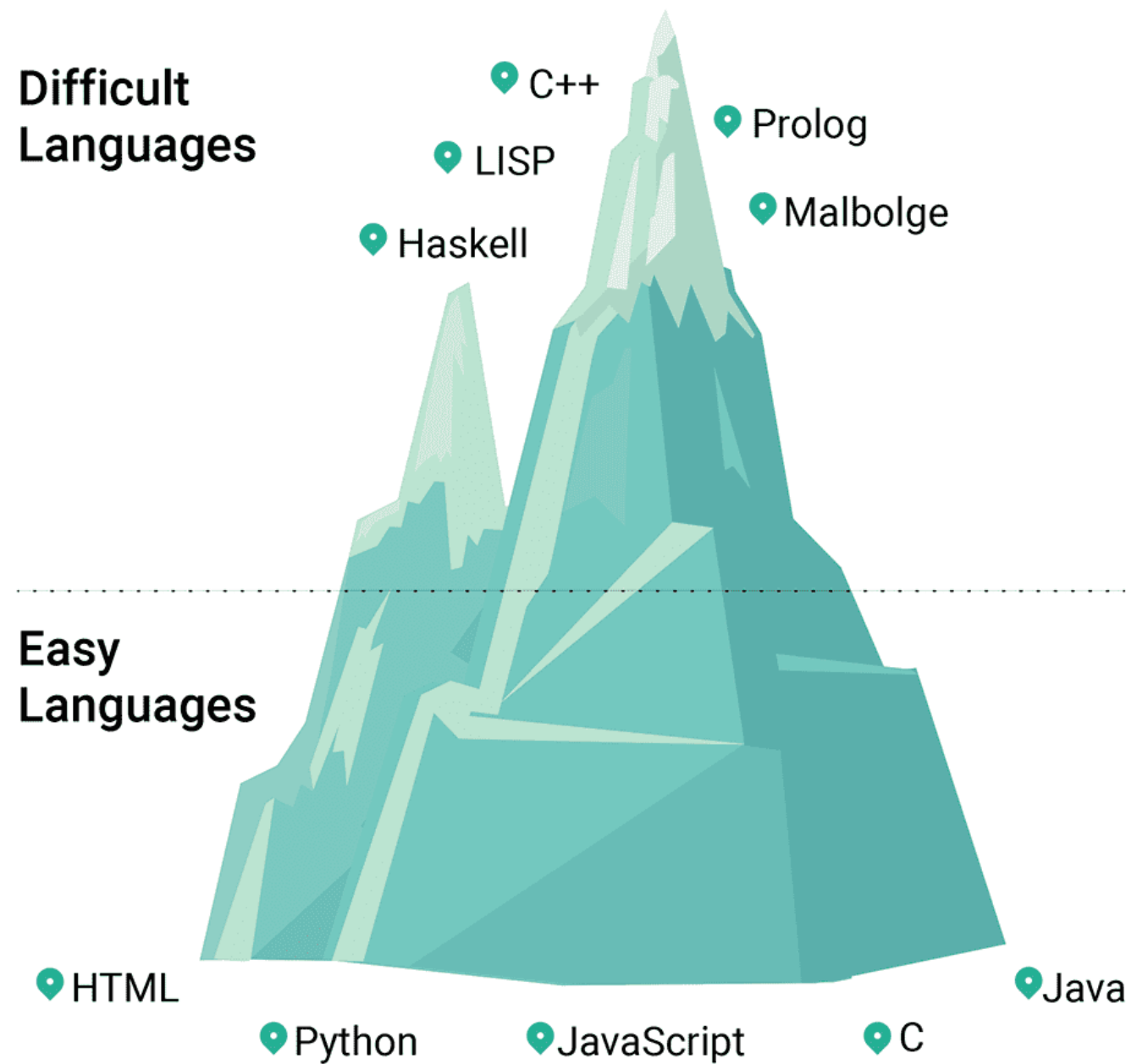
آرایه ها، اشاره گر ها و ...

ورودی و خروجی با فایل ها



Timeline of Programming Languages





Python

زبان برنامه‌نویسی سطح بالا و تفسیری است
نحو ساده و خوانا، مناسب برای توسعه سریع و
مبتدیان

تایپ پویا و مدیریت خودکار حافظه باعث
انعطاف‌پذیری بیشتر می‌شود

کتابخانه‌های غنی و قابلیت‌های چندمنظوره برای
کاربردهای متنوع

عملکرد کندتر نسبت به C به دلیل ماهیت
تفسیری

کنترل کمتر بر روی منابع سیستم و خطرات
بیشتر ناشی از تایپ پویا

high-level

زبان برنامه‌نویسی سطح پایین و کامپایلی است
عملکرد بسیار سریع و کارآمد برای برنامه‌نویسی
سیستمی

امکان کنترل دقیق بر روی حافظه و منابع
سیستم را فراهم می‌کند

نحو پیچیده‌تر و مدیریت دستی حافظه باعث
می‌شود برای مبتدیان دشوارتر باشد

خطرات بیشتری مانند نشتی حافظه و سرریز بافر
وجود دارد

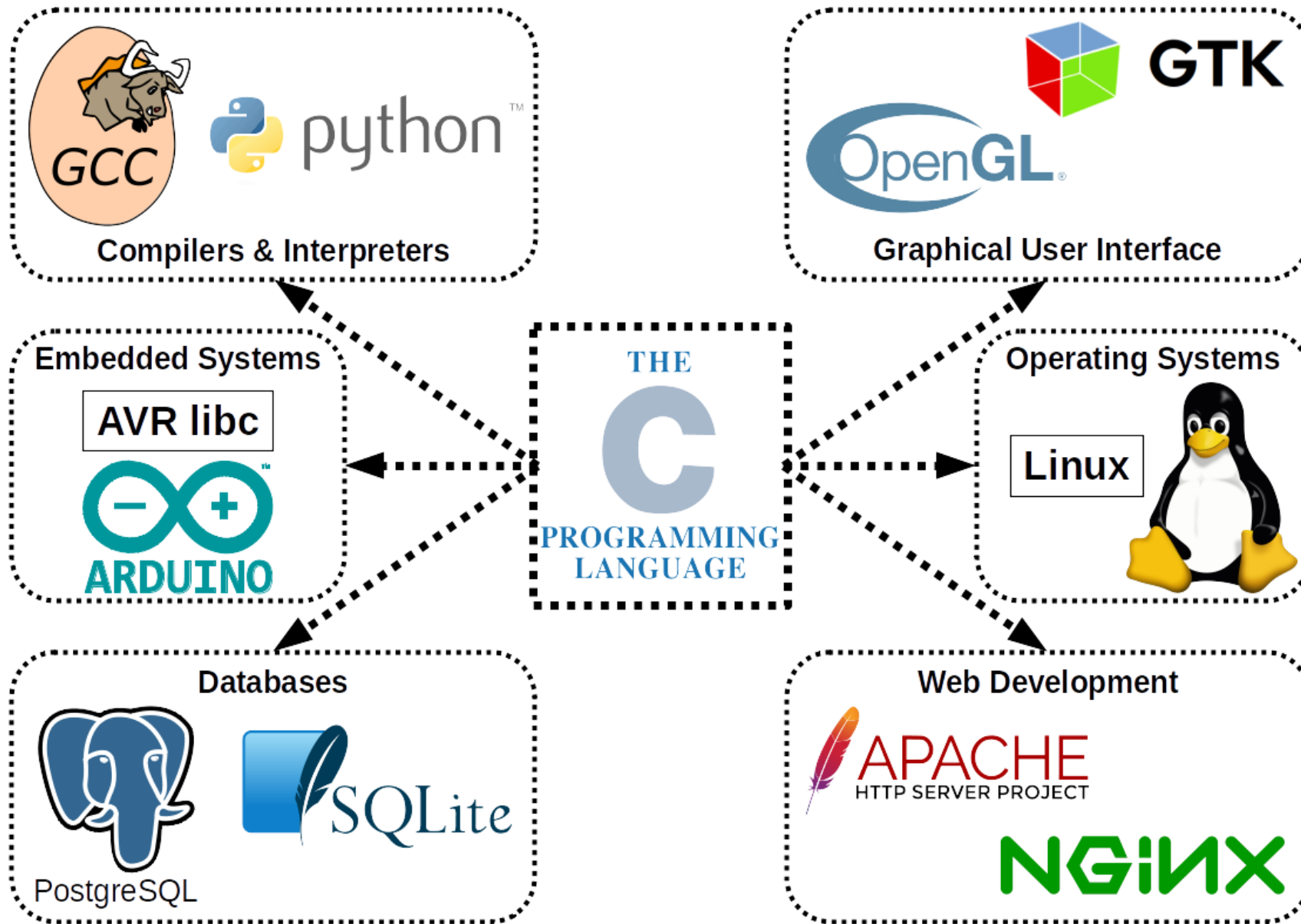
قابلیت حمل بالا و پایه بسیاری از زبان‌های مدرن
است

mid-level

چرا زبان C؟

- تعریف C: یک زبان برنامه‌نویسی همه‌منظوره و رویه‌ای است.
- موارد استفاده: برنامه‌نویسی سیستم‌ها، سیستم‌های تعبیه‌شده، توسعه بازی و غیره.
- اهمیت: پایه‌ای برای بسیاری از زبان‌های مدرن مانند C++، جاوا و پایتون.





تاریخچه زبان C

- توسعه‌دهنده: دنیس ریچی در آزمایشگاه‌های بل.
- سال: ۱۹۷۲.
- هدف: ایجاد برای توسعه سیستم عامل UNIX
- تکامل: تأثیرگذار بر بسیاری از زبان‌های بعدی.



ویژگی های C

مزایای استفاده از زبان برنامه نویسی C توضیح داده شده است:

1. سیستم عامل ها: C به دلیل قابلیت های بالا و قابلیت پورتینگ، برای پیاده سازی سیستم عامل های مختلف مانند لینوکس، ویندوز و macOS مناسب است.
2. سیستم های تعبیه شده: بیشتر میکروپردازنده های تولید شده هر سال در دستگاه های غیر از کامپیوترهای چندمنظوره تعبیه می شوند. C یکی از محبوب ترین زبان های برنامه نویسی برای توسعه این سیستم های تعبیه شده است.
3. سیستم های زمان واقعی: C برای برنامه نویسی سیستم های زمان واقعی که نیاز به پاسخدهی سریع و قابل پیش بینی دارند، مناسب است.
4. سیستم های ارتباطی: C برای توسعه سیستم های ارتباطی که نیاز به انتقال سریع و بدون تأخیر داده های صوتی و تصویری دارند، مناسب است. به طور کلی، C به دلیل قابلیت های بالا در پردازش و کنترل سخت افزار، یکی از محبوب ترین زبان های برنامه نویسی برای توسعه سیستم های با عملکرد بالا است.



مفاهیم اولیه

همگردان یا کامپایلر برنامه یا مجموعه‌ای از برنامه‌های کامپیوتری است که متنی از زبان برنامه نویسی سطح بالا (زبان مبدا) را به زبانی سطح پایین (زبان مقصد)، مثل اسمبلی یا زبان ماشین، تبدیل می‌کند. خروجی این برنامه ممکن است برای پردازش شدن توسط برنامه دیگری مثل پیوند دهنده مناسب باشد یا فایل متنی باشد که انسان نیز بتواند آن را بخواند.

یک **IDE** یا به طور کامل **محیط توسعه یکپارچه** که مخففی از **integrated development environment** می باشد. برنامه نرم افزاری است که برای کمک به برنامه نویسان و توسعه دهندگان جهت ساخت نرم افزار طراحی شده است. اکثر IDEها شامل یک ویرایشگر کد منبع، یک یا چند کامپایلر و یک اصلاح کننده خطا میباشند.

کامپایلر وظیفه ی تبدیل کد های برنامه نویسی به زبان قابل فهم ماشین را برعهده دارد اما IDE یک نرم افزار کمکی برای راحتتر شدن برنامه نویسی است. بی شک زبان C و ++C جزء قدرتمندترین و مشهورترین زبان های برنامه نویسی جهان هستند و کامپایلر ها و IDE های بسیاری برای آن ها عرضه شده است. که تعداد محدودی از آن ها دارای محبوبیت و قدرت کافی هستند



کامپایلر های محبوب زبان C و C++

1. **MinGw**: نرم افزار MinGw کامپایلر مخصوص مایکروسافت میباشد که فقط از ویندوز پشتیبانی میکند و برای C RunTime و برخی دیگر از زبان های RunTime میباشد.
2. **GCC**: نرم افزار GCC یک کامپایلر رایگان زیر نظر GNU میباشد که نه تنها کد های C++ - C را کامپایل میکند بلکه از زبان های Ada , Objective_C , java نیز پشتیبانی میکند.
3. **Tiny C Compiler**: نرم افزار TCC یکی از بهترین کامپایلر های C میباشد که از تمامی پیش پردازنده ها پشتیبانی میکند و در آن از اسمبلر GNU استفاده شده است. لازم به ذکر است که اسمبلر GNU یکی از بهترین اسمبلر های جهان است.
4. **Ideone**: نرم افزار Ideone یک IDE و کامپایلر آنلاین میباشد که از C++ - C و ۶۰ زبان دیگر پشتیبانی میکند.



معرفی انواع IDE کاربردی در زبان C و C++

Visual studio: از قابلیت های VS میتوان به برنامه نویسی برای موبایل ، وب و دکستاپ اشاره کرد و پشتیبانی از زبان های بسیاری هم چون C# , Basic , Asp.net , Python , C , C++ , Css , Xml , Ruby , JavaScript و هم چنین قابلیت های بیشتر دیگر اما از بدی های آن میتوان پشتیبانی نکردن از دیگرسystem عامل ها و کامپایلر ها ، حجم بسیار زیاد و قیمت سرسام آور آن اشاره کرد.

Code::Blocks: یک ادیتور مخصوص C-C++ است که البته در نگارش جدید آن Fortran نیز اضافه شده است سرعت بالا پشتیبانی از تمام سیستم عامل ها ، کامپایلرها ، حجم بسیار کم و همچنین رایگان و متن باز (Open Surce) بودن آن ، آن را در بین برنامه نویسان بسیار محبوب کرده است .

Kdevelop: ادیتور Kd یک ادیتور C++ - C رایگان متن باز و کم حجم برای سیستم عامل های خانواده ی لینوکس و Mac میباشد . این ادیتور از فریم ورک قدرتمند Qt نیز پشتیبانی میکند و البته نسخه های مختلفی از آن برای پشتیبانی از زبان های Php و Python نیز ارائه شده است . از بدی های این ادیتور میتوان پشتیبانی نکردن از سیستم عامل محبوب ویندوز نام برد .



IDEs!

- ACK
- Borland Turbo C
- Clang
- GCC
- ICC
- LCC
- Norcroft C
- PCC
- SDCC
- TCC
- Visual Studio,



1. Code::Blocks

- Platforms: Windows, Mac, Linux
- Features: Customizable interface, support for multiple compilers, and a powerful debugger.

2. Eclipse CDT

- Platforms: Windows, Mac, Linux
- Features: Extensive plugin support, integrated debugging, and project management tools.

3. CLion

- Platforms: Windows, Mac, Linux
- Features: Smart code completion, powerful refactoring tools, and integrated debugger. (Paid)

4. Visual Studio

- Platforms: Windows
- Features: Excellent debugging tools, GUI design capabilities, and a rich set of extensions. (Community version available for free)

5. Xcode

- Platforms: Mac
- Features: Integrated development environment for macOS with a suite of software development tools.

6. NetBeans

- Platforms: Windows, Mac, Linux
- Features: Supports various languages, including C, with a simple interface and good debugging tools.

7. Atom

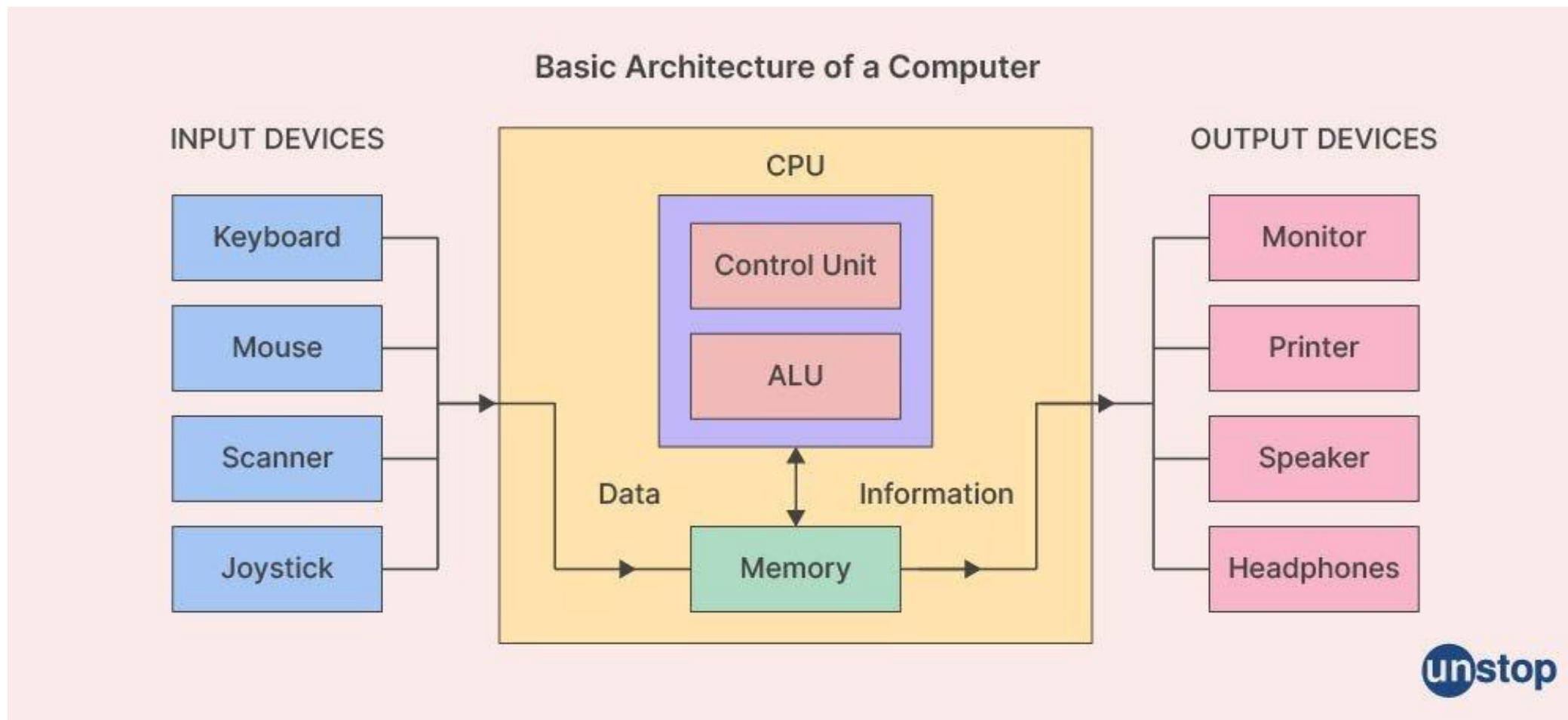
- Platforms: Windows, Mac, Linux
- Features: Highly customizable with packages for C programming, lightweight, and user-friendly.

8. Geany

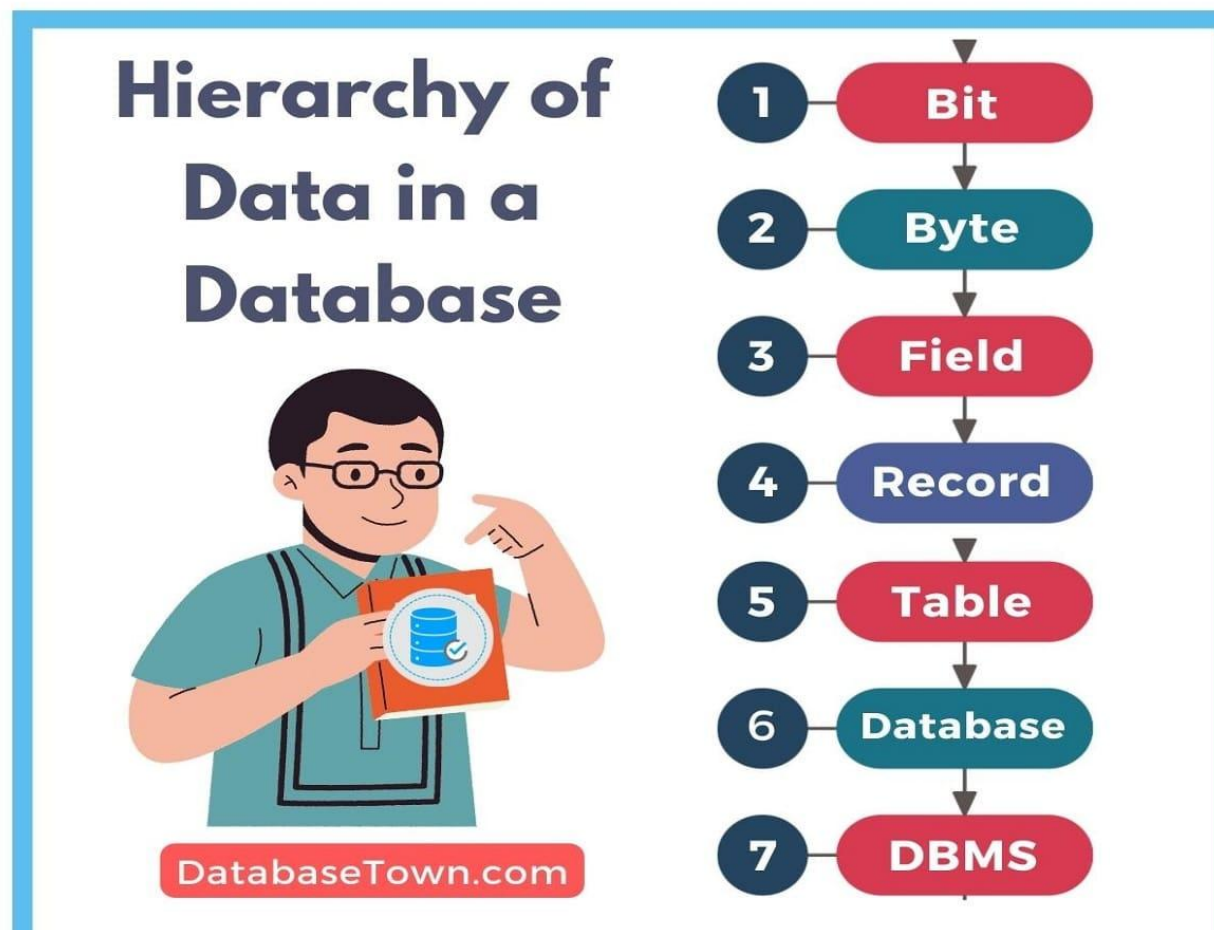
- Platforms: Windows, Mac, Linux
- Features: Lightweight and fast, with basic IDE features and support for multiple programming languages.



معماری کامپیوتر



سلسله مراتب داده

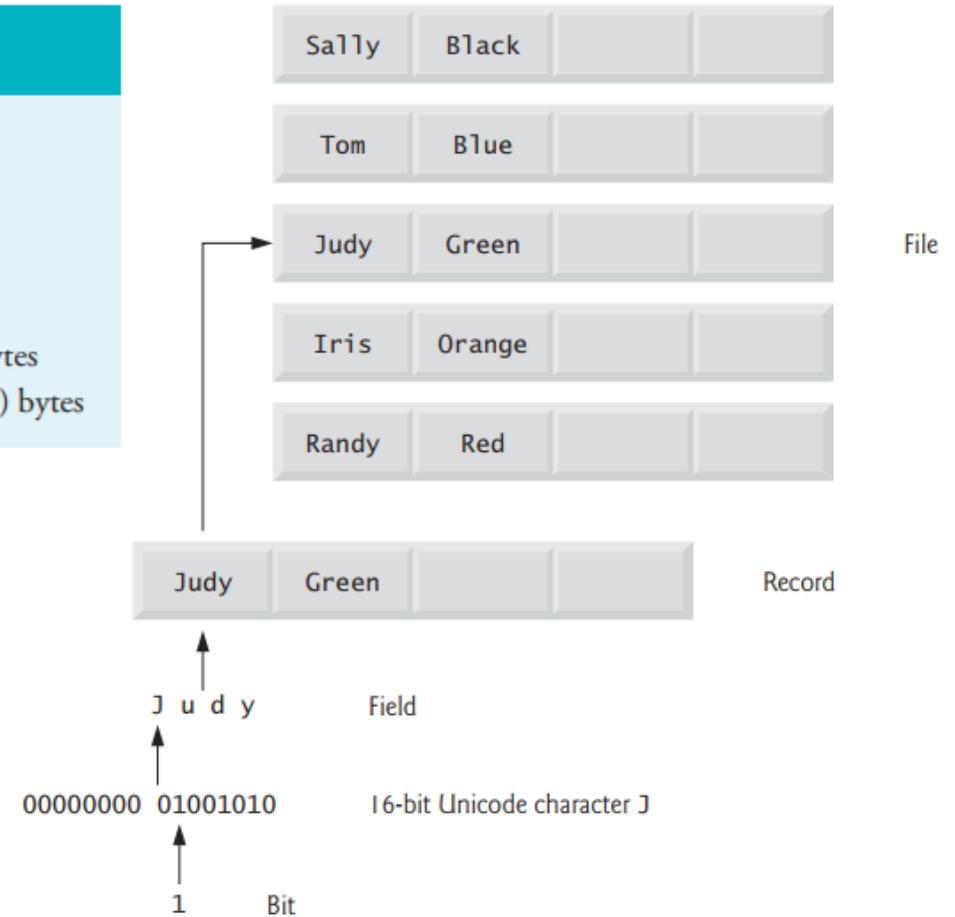


مدیریت پایگاه داده (DBMS) نرم‌افزاری است که مدیریت و ذخیره‌سازی داده‌ها در پایگاه داده را برعهده دارد و به کاربران امکان تعامل با داده‌ها را از طریق دستورات SQL (Structured Query Language) می‌دهد.

Database management system

| Unit | Bytes | Which is approximately |
|------------------|----------------|---|
| 1 kilobyte (KB) | 1024 bytes | 10^3 bytes (1024 bytes exactly) |
| 1 megabyte (MB) | 1024 kilobytes | 10^6 (1,000,000) bytes |
| 1 gigabyte (GB) | 1024 megabytes | 10^9 (1,000,000,000) bytes |
| 1 terabyte (TB) | 1024 gigabytes | 10^{12} (1,000,000,000,000) bytes |
| 1 petabyte (PB) | 1024 terabytes | 10^{15} (1,000,000,000,000,000) bytes |
| 1 exabyte (EB) | 1024 petabytes | 10^{18} (1,000,000,000,000,000,000) bytes |
| 1 zettabyte (ZB) | 1024 exabytes | 10^{21} (1,000,000,000,000,000,000,000) bytes |

8bit: 1byte



سیستم‌های عددی در C

سیستم‌های عددی در C:

- از سه سیستم عددی اصلی پشتیبانی می‌کند:

1. **دهدهی (Decimal)** اعداد معمولی که از ارقام ۰ تا ۹ استفاده می‌کنند.

• مثال 42, 3.14 :

2. **دودویی (Binary)** اعداد با ارقام ۰ و ۱.

• مثال 0b101010, 0b11.001 :

3. **شانزدهدهی (Hexadecimal)** اعداد با ارقام ۰ تا ۹ و حروف A تا F.

• مثال 0x2A, 0x3.3333 :

- در C می‌توان از این سیستم‌های عددی برای تعریف متغیرها، انجام محاسبات و دستکاری داده‌ها استفاده کرد.

ASCII در C:

- ASCII (American Standard Code for Information Interchange) یک سیستم کدگذاری استاندارد برای نمایش حروف، اعداد و نمادهاست.

- در C، ASCII کاربردهای متنوعی دارد:

- تعریف و استفاده از کاراکترها در متغیرها و رشته‌ها

- انجام عملیات مقایسه و محاسبه روی کاراکترها

- خواندن و نوشتن کاراکترها در ورودی/خروجی

- استفاده از کدهای ASCII برای کنترل خروجی (مثل کنترل نمایش)

به طور خلاصه، سیستم‌های عددی و ASCII در C به توسعه‌دهندگان امکان دستکاری و کار با اعداد و کاراکترها را می‌دهد و در بسیاری از برنامه‌نویسی‌ها کاربرد دارد.

| Binary digit | Octal digit | Decimal digit | Hexadecimal digit |
|--------------|-------------|---------------|-------------------------|
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| | 2 | 2 | 2 |
| | 3 | 3 | 3 |
| | 4 | 4 | 4 |
| | 5 | 5 | 5 |
| | 6 | 6 | 6 |
| | 7 | 7 | 7 |
| | | 8 | 8 |
| | | 9 | 9 |
| | | | A (decimal value of 10) |
| | | | B (decimal value of 11) |
| | | | C (decimal value of 12) |
| | | | D (decimal value of 13) |
| | | | E (decimal value of 14) |
| | | | F (decimal value of 15) |

Fig. C.1 | Digits of the binary, octal, decimal and hexadecimal number systems.

| Attribute | Binary | Octal | Decimal | Hexadecimal |
|---------------|--------|-------|---------|-------------|
| Base | 2 | 8 | 10 | 16 |
| Lowest digit | 0 | 0 | 0 | 0 |
| Highest digit | 1 | 7 | 9 | F |

Fig. C.2 | Comparing the binary, octal, decimal and hexadecimal number systems.

| Positional values in the decimal number system | | | |
|--|----------|--------|--------|
| Decimal digit | 9 | 3 | 7 |
| Position name | Hundreds | Tens | Ones |
| Positional value | 100 | 10 | 1 |
| Positional value as a power of the base (10) | 10^2 | 10^1 | 10^0 |

Fig. C.3 | Positional values in the decimal number system.

| Positional values in the binary number system | | | |
|---|-------|-------|-------|
| Binary digit | 1 | 0 | 1 |
| Position name | Fours | Twos | Ones |
| Positional value | 4 | 2 | 1 |
| Positional value as a power of the base (2) | 2^2 | 2^1 | 2^0 |

Fig. C.4 | Positional values in the binary number system.

| Positional values in the octal number system | | | |
|--|-------------|--------|-------|
| Decimal digit | 4 | 2 | 5 |
| Position name | Sixty-fours | Eights | Ones |
| Positional value | 64 | 8 | 1 |
| Positional value as a power of the base (8) | 8^2 | 8^1 | 8^0 |

Fig. C.5 | Positional values in the octal number system.

| Positional values in the hexadecimal number system | | | |
|--|-----------------------------|----------|--------|
| Decimal digit | 3 | D | A |
| Position name | Two-hundred-and-fifty-sixes | Sixteens | Ones |
| Positional value | 256 | 16 | 1 |
| Positional value as a power of the base (16) | 16^2 | 16^1 | 16^0 |

Fig. C.6 | Positional values in the hexadecimal number system.

| Decimal number | Binary representation | Octal representation | Hexadecimal representation |
|----------------|-----------------------|----------------------|----------------------------|
| 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 |
| 2 | 10 | 2 | 2 |
| 3 | 11 | 3 | 3 |
| 4 | 100 | 4 | 4 |
| 5 | 101 | 5 | 5 |
| 6 | 110 | 6 | 6 |
| 7 | 111 | 7 | 7 |
| 8 | 1000 | 10 | 8 |
| 9 | 1001 | 11 | 9 |
| 10 | 1010 | 12 | A |
| 11 | 1011 | 13 | B |
| 12 | 1100 | 14 | C |
| 13 | 1101 | 15 | D |
| 14 | 1110 | 16 | E |
| 15 | 1111 | 17 | F |
| 16 | 10000 | 20 | 10 |

Fig. C.7 | Decimal, binary, octal and hexadecimal equivalents.

ASCII character set

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 0 | nul | soh | stx | etx | eot | enq | ack | bel | bs | ht |
| 1 | lf | vt | ff | cr | so | si | dle | dc1 | dc2 | dc3 |
| 2 | dc4 | nak | syn | etb | can | em | sub | esc | fs | gs |
| 3 | rs | us | sp | ! | " | # | \$ | % | & | ' |
| 4 | (|) | * | + | , | - | . | / | 0 | 1 |
| 5 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; |
| 6 | < | = | > | ? | @ | A | B | C | D | E |
| 7 | F | G | H | I | J | K | L | M | N | O |
| 8 | P | Q | R | S | T | U | V | W | X | Y |
| 9 | Z | [| \ |] | ^ | _ | ' | a | b | c |
| 10 | d | e | f | g | h | i | j | k | l | m |
| 11 | n | o | p | q | r | s | t | u | v | w |
| 12 | x | y | z | { | | } | ~ | del | | |

Fig. B.1 | ASCII Character Set.

The digits at the left of the table are the left digits of the decimal equivalent (0–127) of the character code, and the digits at the top of the table are the right digits of the character code. For example, the character code for “F” is 70, and the character code for “&” is 38.

2

INTRO. TO PROGRAMMING



زبان‌های برنامه‌نویسی

زبان‌های ماشینی هر کامپیوتری تنها می‌تواند زبان ماشینی خود را که توسط معماری سخت‌افزاری آن تعریف شده است، به طور مستقیم درک کند. زبان‌های ماشینی معمولاً از اعداد (در نهایت به ۱ و ۰ تقلیل یافته) تشکیل شده‌اند. چنین زبان‌هایی برای انسان‌ها دشوار و ناراحت‌کننده هستند.

زبان‌های اسمبلی برنامه‌نویسی به زبان ماشینی برای اکثر برنامه‌نویسان بسیار کند و خسته‌کننده بود. به جای آن، آنها شروع به استفاده از مخفف‌های شبیه به انگلیسی برای نمایش عملیات ابتدایی کردند. این مخفف‌ها پایه و اساس زبان‌های اسمبلی را تشکیل دادند. برنامه‌های ترجمان به نام «اسمبلرها» توسعه یافتند تا کد زبان اسمبلی را به زبان ماشینی تبدیل کنند. اگرچه کد زبان اسمبلی برای انسان‌ها روشن‌تر است، اما تا زمان ترجمه به زبان ماشینی، برای کامپیوترها غیرقابل فهم است.

زبان‌های سطح بالا برای افزایش سرعت فرآیند برنامه‌نویسی، زبان‌های سطح بالا توسعه یافتند که در آن می‌توان با نوشتن یک دستور، به انجام وظایف قابل توجهی پرداخت. زبان‌های سطح بالا به شما امکان می‌دهند دستوراتی بنویسید که تقریباً شبیه به انگلیسی روزمره و حاوی عبارات ریاضی رایج هستند. برنامه‌های ترجمان به نام «کامپایلرها» زبان‌های سطح بالا را به زبان ماشینی تبدیل می‌کنند.

فرآیند کامپایل یک برنامه سطح بالا به زبان ماشینی ممکن است مقدار قابل توجهی زمان کامپیوتر را در بر بگیرد. برنامه‌های «مفسر» توسعه یافتند تا بتوانند برنامه‌های زبان سطح بالا را مستقیماً (بدون نیاز به کامپایل) اجرا کنند، هرچند با سرعت کمتری نسبت به برنامه‌های کامپایل شده. زبان‌های اسکریپتی مانند JavaScript و PHP توسط مفسرها پردازش می‌شوند.



Programming Languages

- There are three types of programming Languages

1) Machine Languages (machine codes):

- Strings of **1s and 0s**.
- Only understood by **integrated circuits**, such as microprocessors.

Example: 10100010
 01011011
 10101010

2) Assembly Languages:

- English-like **abbreviations** representing elementary computer operations.
- translated to machine code by using **assemblers**.

Example: MOV AL, 3BH
 ADD AL, AH
 SUB AL, AH
 MOV [SI]



Programming Languages

- There are three types of programming Languages

1) Machine Language

2) Assembly Languages

3) High-level Languages:

- Codes similar to everyday English
- Use mathematical notations
- translated to machine code by using compilers.
- C, C++, PASCAL, FORTRAN, BASIC are high-level languages.

Example:

```
c=a+b; if(a<b)
    printf("a is less than b\n");
else
    printf("a is NOT less than b\n");
```



Structured programming

- Disciplined approach to writing programs
 - Using flowcharts (graphical representation)
 - Using pseudocodes or step by step algorithms.
- Clear, easy to test and debug and easy to modify
 - Using functions for efficient programming.

Multitasking

Specifying that many activities run in parallel.



برنامه‌نویسی ساختاریافته

رویکرد انضباطی برای نوشتن برنامه‌ها است که شامل موارد زیر است:

۱. استفاده از نمودارهای جریان (نمایش گرافیکی) : نمودارهای جریان برای ترسیم گرافیکی الگوریتم‌ها و ساختار برنامه استفاده می‌شوند.

۲. استفاده از شبه‌کدها یا الگوریتم‌های مرحله به مرحله: شبه‌کدها برای توصیف الگوریتم‌ها به صورت گام‌به‌گام استفاده می‌شوند.

۳. ایجاد کد واضح، آسان برای آزمایش و اشکال‌زدایی و تغییر: این رویکرد باعث ایجاد کد قابل خواندن، قابل تست و قابل تغییر می‌شود.

۴. استفاده از توابع برای برنامه‌نویسی کارآمد: تابع‌ها برای کپسوله‌سازی و استفاده مجدد از کد استفاده می‌شوند.

چندوظیفگی

مشخص کردن اینکه چندین فعالیت به صورت موازی اجرا شوند.

در مجموع، برنامه‌نویسی ساختاریافته رویکردی منظم و سازمان‌یافته برای توسعه برنامه‌های کامپیوتری است که باعث ایجاد کد با کیفیت بالا، قابل تست و قابل تغییر می‌شود.



Basics of a Typical C Program Development Process

- Phases of C Programs:**

1. **Edit**

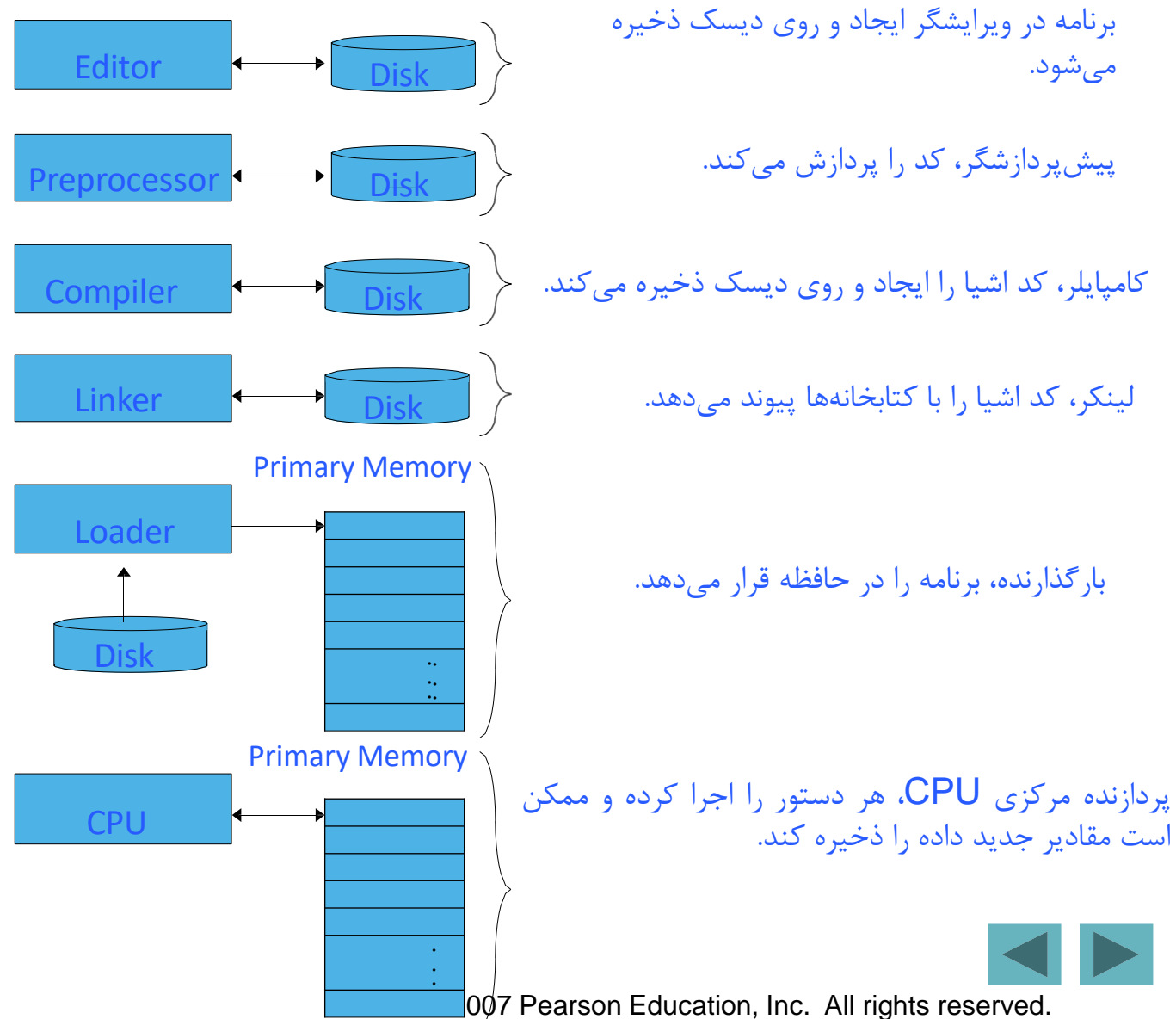
2. **Preprocess**

3. **Compile**

4. **Link**

5. **Load**

6. **Execute**



Simple C Program:

- The following program displays "Hello World" on the computer screen (monitor).

```
/* This is our first program in C Language */  
#include <stdio.h>  
int main()  
{  
    printf("Hello World\n");  
    return 0;  
}
```

- The program output

```
Hello World
```



Simple C Program:

```
/* This is our first program in C Language */
#include <stdio.h>
int main()
{
    printf("Hello World\n");
    return 0;
}
```

Comments:

- Text surrounded by `/*` and `*/` is ignored by computer.
- Used to describe program.

`#include <stdio.h>`

دستور پیش پردازنده:

این یک دستور برای کامپیوتر است. به کامپیوتر می گوید که باید فایل به نام "stdio.h" را پیدا کند و محتویات آن را به برنامه اضافه کند. فایل stdio.h شامل دستورات و توابعی است که برای کار با ورودی و خروجی استفاده می شوند. مثلاً دستور printf() که برای چاپ چیزی روی صفحه استفاده می شود، در این فایل قرار دارد.



Simple C Program:

```
/* This is our first program in C Language */  
#include <stdio.h>  
int main()  
{  
    printf("Hello World\n");  
    return 0;  
}
```

`int main()`

"main()" تابع اصلی و
ضروری در هر برنامه سی است

- برنامه‌های C شامل یک یا چند تابع هستند.
- یکی از این توابع باید با نام `main()` باشد.
- استفاده از پرانتز "()" نشان می‌دهد که این یک تابع است.
- `int` یعنی که تابع `main()` یک مقدار عددی صحیح (integer) را به عنوان خروجی برمی‌گرداند.
- دو آکولاد "{ }" مشخص می‌کنند که بدنه یا محتویات یک بلوک کد است.
- بدنه همه توابع باید درون آکولادها قرار بگیرد.



Simple C Program:

```
/* This is our first program in C Language */
#include <stdio.h>
int main()
{
    printf("Hello World\n");
    return 0;
}
```

- printf() رشته کاراکترهای داخل گیومه (" ") را چاپ می کند.
 - دستورات باید با نقطه ویرگول (;) تمام شوند
 - \n برای ایجاد خط جدید استفاده می شود
 - return 0 نشان می دهد برنامه به طور عادی اجرا شده است
 - آکولاد بسته شده نشان می دهد که تابع main به پایان رسیده است
- Right brace }**



Simple C Program:

- **مثال ۱:** برنامه‌ای که نام و نام خانوادگی تان را در دو خط متوالی نمایش دهد:

- **مثال ۲:** برنامه‌ای که خطوط زیر را نمایش دهد:

```
Today  
is a nice  
day
```



C Program: Addition of two integer numbers

```
/* This program adds two integer numbers */
#include <stdio.h>

int main()
{
    int a, b, sum;          /* variable declarations */
    printf("Enter first integer\n"); /* prompt the user */
    scanf( "%d", &a);        /* read first integer */
    printf("Enter second integer\n"); /* prompt the user */
    scanf( "%d", &b);        /* read second integer */
    sum = a + b;            /* calculate the sum */
    printf( "Sum = %d\n", sum ); /* print the calculated sum*/
    return 0; /* indicate that program ended successfully */
}
```

```
Enter first integer
15
Enter second integer
26
Sum = 41
```

Program Output



C Program: Addition of two integer numbers

```
int a, b, sum;
```

- این خط، متغیرهایی به نام‌های a، b و sum را تعریف می‌کند.
 - **متغیرها:** محل‌هایی در حافظه هستند که مقادیر می‌توانند در آن‌ها ذخیره شوند. int به این معنی است که این متغیرها می‌توانند اعداد صحیح (-۱، ۳، ۰، ۴۷) را نگه دارند.
 - نام‌های متغیرها (شناسه‌ها): Variable names (identifiers)
 - `a, b, sum;`
- شناسه‌ها می‌توانند شامل حروف، اعداد (اما نباید با عدد شروع شوند) و زیرخط (_) باشند. همچنین به حروف بزرگ و کوچک حساس هستند.
- Declarations appear before executable statements
 - If an executable statement references and undeclared variable it will produce a syntax (compiler) error.

تعریف متغیرها باید قبل از استفاده از آن‌ها انجام شود



C Program: Addition of two integer numbers

```
scanf ( "%d" , &a ) ;
```

- Obtains(reads/inputs) a value from the user
 - **scanf** uses standard input (usually keyboard)
- This **scanf** statement has two arguments
 - %d** - indicates data should be a decimal integer
 - &a** – location (address) in memory to store variable **a**.

& is confusing in beginning – for now, just remember to include it with the variable name in **scanf** statements.
- When executing the program the user responds to the **scanf** statement by typing in a number, then pressing the *enter* (return) key.

• scanf() برای دریافت ورودی از کاربر استفاده می شود
• %d نشان می دهد که داده باید عدد صحیح باشد
• &a آدرس متغیر a را مشخص می کند تا مقدار در آن ذخیره شود
• کاربر باید پس از دیدن scanf() یک عدد را تایپ و Enter را فشار دهد



C Program: Addition of two integer numbers

= (assignment operator)

- Assigns a value to a variable
- Is a binary operator (has two operands)

```
sum = a + b;
```

sum gets a + b;

- Variable receiving value on left

```
printf( "Sum is %d\n", sum );
```

- Similar to `scanf`
 - `%d` means decimal integer will be printed
 - `sum` specifies what integer will be printed
- Calculations can be performed inside `printf` statements

```
printf( "Sum is %d\n", a + b );
```



C Program: Addition of two integer numbers

مثال ۳: یک برنامه C بنویسید که جمع اعداد صحیح ۷، ۸ و ۱۴ را محاسبه و نمایش دهد.

مثال ۴: یک برنامه C بنویسید که از کاربر بخواهد ۳ عدد صحیح وارد کند و مجموع این سه عدد را خروجی دهد.



Arithmetic Operations in C

- Arithmetic Calculations:**

- Use ***** for multiplication and **/** for division
- Integer division truncates remainder
 - $7 / 5$ evaluates to **1**
- Modulus operator(**%**) returns the remainder of modular division.

$7 \% 5$ evaluates to **2**

- Operator precedence:**

- Some arithmetic operators act before others (i.e., multiplication before addition)
- Use parenthesis when needed
- Example: Find the average of three variables a, b and c

Do not use: $a + b + c / 3$

Use: $(a + b + c) / 3$

محاسبات حسابی:

• از ***** برای ضرب و **/** برای تقسیم استفاده کنید.

• تقسیم صحیح باقی مانده را حذف می کند.

• $7 / 5$ برابر است با **1**.

• اپراتور مدول **%** باقی مانده تقسیم را برمی گرداند.

• $7 \% 5$ برابر است با **2**.

اولویت عملگرها:

• برخی از عملگرهای حسابی قبل از بقیه عمل می کنند (مثلاً ضرب قبل از جمع).

• در صورت نیاز از پرانتز استفاده کنید.



Arithmetic Operations in C

- Arithmetic operators:

| C operation | Arithmetic operator | Algebraic expression | C expression |
|----------------|---------------------|----------------------|--------------------|
| Addition | + | $f + 7$ | <code>f + 7</code> |
| Subtraction | - | $p - c$ | <code>p - c</code> |
| Multiplication | * | bm | <code>b * m</code> |
| Division | / | x / y | <code>x / y</code> |
| Modulus | % | $r \text{ mod } s$ | <code>r % s</code> |



Arithmetic Operations in C

- Rules of Operator Precedence:**

| Operator(s) | Operation(s) | Order of evaluation (precedence) |
|-------------|-----------------------------------|--|
| () | Parentheses | Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses “on the same level” (i.e., not nested), they are evaluated left to right. |
| *, /, or % | Multiplication, Division, Modulus | Evaluated second. If there are several, they are evaluated left to right. |
| + or - | Addition Subtraction | Evaluated last. If there are several, they are evaluated left to right. |



Decision Making: **Equality** and **Relational Operators**

- **Executable statements**
 - Perform actions (calculations, input/output of data)
 - Perform decisions
 - May want to print "**pass**" or "**fail**" given the value of a test grade
- **if control structure**
 - Simple version in this section, more detail later
 - If a condition is **true**, then the body of the **if** statement executed
 - **0** is **false**, non-zero is **true**
 - Control always resumes after the **if** structure
- **Keywords**
 - Special words reserved for C
 - Cannot be used as identifiers or variable names



Decision Making: Equality and Relational Operators

| Standard algebraic equality operator or relational operator | C equality or relational operator | Example of C condition | Meaning of C condition |
|---|-----------------------------------|------------------------|---|
| <i>Equality Operators</i> | | | |
| = | == | x == y | x is equal to y |
| not = | != | x != y | x is not equal to y |
| <i>Relational Operators</i> | | | |
| > | > | x > y | x is greater than y |
| < | < | x < y | x is less than y |
| >= | >= | x >= y | x is greater than or equal to y |
| <= | <= | x <= y | x is less than or equal to y |



Decision Making: **Equality** and **Relational Operators**

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



Decision Making: Equality and Relational Operators

```

1  /* Fig. 2.13: fig02 13.c
2     Using if statements, relational
3     operators, and equality operators */
4  #include <stdio.h>
5
6  int main()
7  {
8     int num1, num2;
9
10    printf( "Enter two integers, and I will tell you\n"
11    printf( "the relationships they satisfy: " );
12    scanf( "%d%d", &num1, &num2 );    /* read two
13
14    if ( num1 == num2 )
15        printf( "%d is equal to %d\n", num1, num2 );
16
17    if ( num1 != num2 )
18        printf( "%d is not equal to %d\n", num1, num2 );
19
20    if ( num1 < num2 )
21        printf( "%d is less than %d\n", num1, num2 );
22
23    if ( num1 > num2 )
24        printf( "%d is greater than %d\n", num1, num2 );
25
26    if ( num1 <= num2 )
27        printf( "%d is less than or equal to %d\n",
28        num1, num2 );

```

Program Outline

1. Declare variables

2. Input

2.1 if statements

3. Print



Decision Making: Equality and Relational Operators

Program Outline

```
29
30     if ( num1 >= num2 )
31         printf( "%d is greater than or equal to %d\n",
32                 num1, num2 );
33
34     return 0;    /* indicate program ended successfully */
35 }
```

3.1 Exit main

```
Enter two integers, and I will tell you
the relationships they satisfy: 3 7
3 is not equal to 7
3 is less than 7
3 is less than or equal to 7
```

```
Enter two integers, and I will tell you
the relationships they satisfy: 22 12
22 is not equal to 12
22 is greater than 12
22 is greater than or equal to 12
```

Program Output



Decision Making: **Equality** and **Relational Operators**

- **Example 5:** Write a C program which asks the user to enter two integers, compare them and perform the following actions:
 - if the first value is greater -> add the two numbers,
 - if the second value is greater -> multiply the integers
 - if they are equal -> divide their multiplication with their sum.

