دانشگاه یزد

مبانی برنامه‌ریزی کامپیوتر
(مبانی سرویسی)
Fundamentals of Programming
(Python)

سمانه رستگاری

جلسه 1: مفاهیم اولیه

عناوین

- آشنایی با کلیت درس
  - سیلابس
  - نحوه ارزیابی
- معرفی اجزای اصلی کامپیوتر
- تاریخچه رشد زبان‌های برنامه‌سازی
- تعاریف اولیه
- آشنایی با زبان پایتون
- مراحل ساخت و اجرای یک برنامه

آشنایی با کلیت درس

هدف:

- آشنایی با مبانی برنامه‌سازی کامپیوتر
- ایجاد تفکر الگوریتمی در حل مسائل
- کسب توانایی پیاده‌سازی الگوریتم‌ها به وسیله کامپیوتر
- آشنایی با اصول اولیه نوشتن برنامه‌های ساخت‌یافته و مهندسی‌ساز

مراجع:

- انگلیسی:

Peter Wentworth, Jeffrey Elkner, Allen B. Downey, and Chris Meyers, "**How to Think Like a Computer Scientist, Learning with Python 3**", October 2012, http://openbookproject.net/thinkcs/python/english3e/

- فارسی:

سعید خالقی و علیرضا حقنیا، "از این پس پایتون"، انتشارات کوشامهر

نحوه ارزیابی

- تمرین 3 نمره
- میان ترم 7 نمره
- پایان ترم 10 نمره

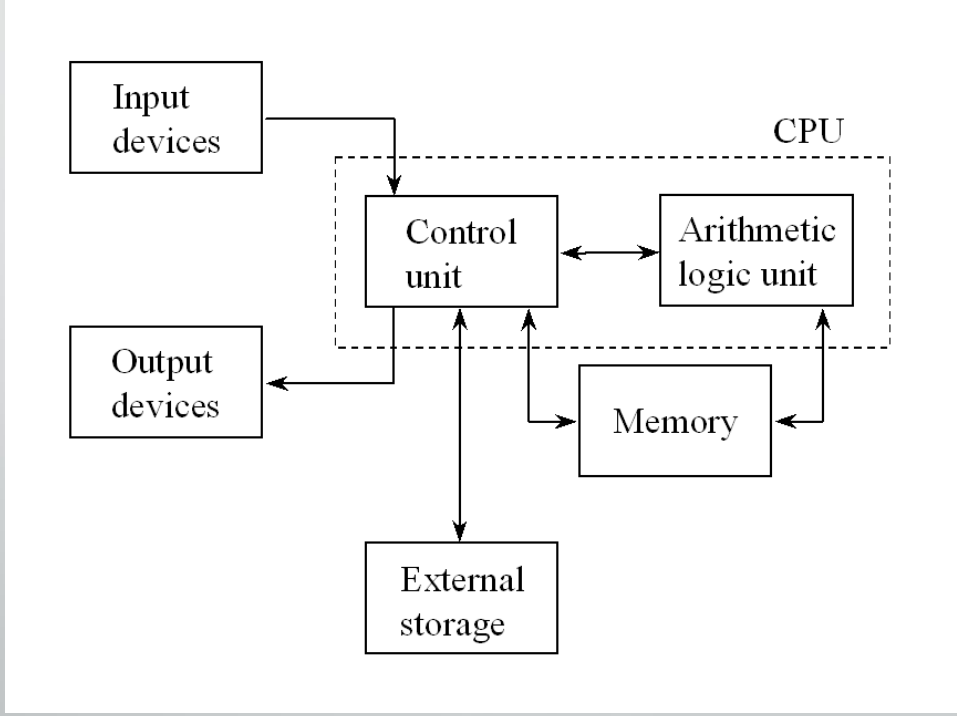
مباحث این جلسه

- معرفی اجزاء کامپیوتر و نقش زبان برنامه‌سازی در آن
- مروری بر مفاهیم الگوریتم
- نوشتن اولین برنامه به زبان Python

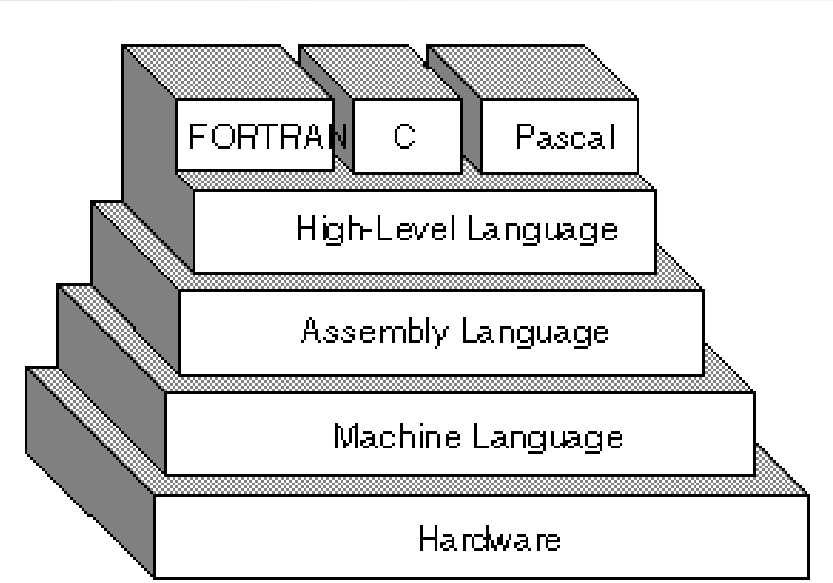
معرفی اجزای اصلی کامپیوتر

- دیدگاه اول:
  1. سخت‌افزار: دستگاه‌هایی که کامپیوتر را می‌سازند مثل واحدهای پردازنده، حافظه، صفحه کلید و ...
  2. نرم‌افزار: برنامه‌هایی که روی کامپیوتر اجرا می‌شوند.
- دیدگاه دوم:
  1. واحدهای ورودی و خروجی
  2. واحد حافظه
  3. واحد محاسبه و منطق (Arithmetic and Logic Unit)
  4. واحد پردازش مرکزی (Central Processing Unit)
  5. واحد ذخیره جانبی

سازماندهی اجزاء کامپیوتر



زبان‌های برنامه‌سازی



کامپیوترها تنها برنامه هایی را می توانند اجرا کنند، که به زبان ماشین تبدیل شده باشند .
بدین سان برنامه هایی که در یک زبان سطح بالا نوشته شده اند، باید قبل از اجرا پردازش شوند و به زبان ماشین ترجمه شوند. این پردازش اضافه مدتی زمان می برد، که این اشکال کوچک زبا نهاي سطح بالا است.

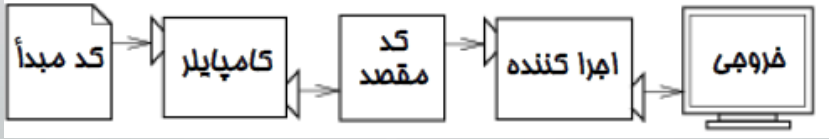
زبان برنامه‌سازی

- زبان برای دستور دادن به دستگاه‌های برنامه‌پذیر مثل کامپیوترها استفاده میشود.
- سیستم‌های عامل، منابع مختلف را مثل CPU، حافظه و... را به برنامه‌های مختلف اختصاص می‌دهند؛ بنابراین واسط بین برنامه‌ها (نرم‌افزارها) و سخت‌افزارها هستند یا در حقیقت مدیریت تقسیم منابع(سخت‌افزار) بین نرم‌افزارها بر عهده سیستم عامل است.
- سیستم‌های عامل، نرم‌افزارها و حتی سخت‌افزارها با استفاده از زبان‌های برنامه‌سازی نوشته و تولید می‌شوند.
- زبان ماشین و یا زبان اسمبلی سطح پایین هستند چون به سخت‌افزار نزیک‌اند.
- زبانی مثل Python یا C سطح بالا هستند و باید قبل از اجرا به چیزی مثل زبان ماشین ترجمه شوند.
- علت وجود زبان‌های سطح بالا: سادگی، کاهش اندازه برنامه، خوانایی بیشتر، اشکال زدایی راحت تر، و قابلیت حمل بیشتر

مفسرها و کامپایلرها

- دو دسته از برنامه ها که زبان هاي سطح بالا را به زبان هاي سطح پایین پردازش می کنند.
- مفسر برنامه را خط به خط می خواند و محاسبات را انجام می دهد.

- یک کامپایلر برنامه را می خواند و قبل از اینکه اجرا کند آن را به طور کامل به زبان ماشین ترجمه می کند.



تعاریف اولیه الگوریتم

- الگوریتم: روش انجام یک کار با **ذکر دقیق تمام مراحل** آن به طوریکه **ترتیب** انجام کارها، **شروع** و **پایان** آن مشخص باشد؛ به عبارت دقیق‌تر هر دستورالعملی که مراحل مختلف کاری را به زبان دقیق و با جزئیات کافی بیان نماید و در آن ترتیب مراحل و خاتمه‌پذیر بودن عملیات کاملاً مشخص باشد الگوریتم نامیده می‌شود.
- فلوچارت: نمایش الگوریتم به صورت شماتیک
- حل مسئله: شامل شناخت مسئله، طرح نقشهٔ حل مسئله و تحلیل راه‌حل مسئله (اجرا، وارسی یا تعمیم راه حل)
- برنامه: نمود الگوریتم به کمک دستورات زبان برنامه‌سازی
- الگوریتم باید بدون ابهام باشد و اجرای مراحل آن بستگی به نظر مجری نداشته باشد.

حل مسئله

- مهارت قدرت شناخت دقیق مسئله (تعیین داده‌ها، مجهول‌ها و رابطه بین آنها)
- مهارت ایجاد طرحی برای حل مسئله به صورت الگوریتمیک (تفکر خلّاقانه در مورد راه‌حل و بیان واضح و دقیق راه‌حل)
- اجرای راه‌حل و اطمینان از درستی راه‌حل

اجزای اصلی الگوریتم

- شروع (دقیقا یک شروع)
- دستورات ورودی/ خروجی: گرفتن مقدار یک عدد، کاراکتر، رشته و ... از کاربر/ نمایش خروجی روی مانیتور یا چاپ روی پرینتر و ...
- دستورات محاسباتی، مقداردهی: انجام محاسبه مشخص، ذخیره کردن یک مقدار در یک متغیر
- دستورات تصمیم‌گیری: ارزیابی یک شرط ساده یا مرکب
- پرش
- پایان (دقیقا یک پایان)

«مقدارx» را از کاربر بگیر.

«پیامx» را به کاربر نشان بده.

«مقدارx» را در «متغیرx» بریز.

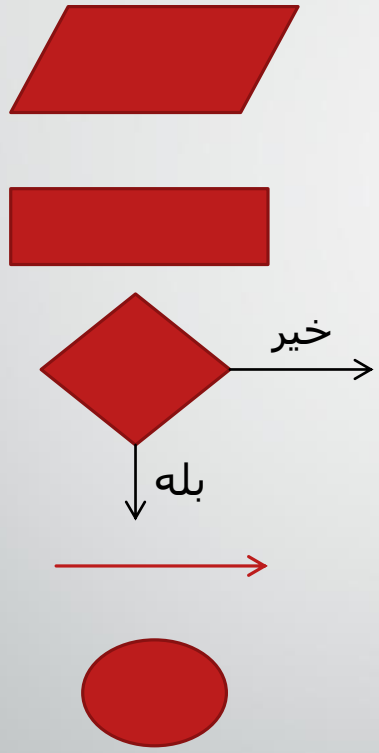
اگر «شرطx» آنگاه «دستورx» وگرنه «دستورy».

برو به «گامx» .

الگوریتم حاصل جمع دو عدد

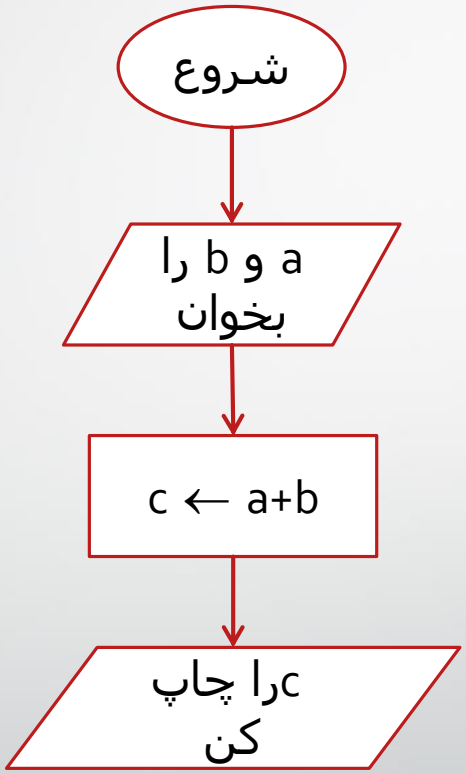
1. شروع کن.
2. عدد a و عدد b را از کاربر بگیر.
3. a را با b جمع کن و حاصل آن را در c بریز.
4. c را چاپ کن.
5. پایان.

فلوچارت



- دستورات ورودی/ خروجی
- مقداردهی و محاسبات
- تصمیم‌گیری
- پرش
- شروع و پایان

فلوچارت حاصل جمع دو عدد



آشنایی بیشتر با زبان Python

- زبان پایتون بسیار ساده است و برای رشته‌های غیر از کامپیوتر مناسب.
- پایتون یک زبان تفسیري معرفی شده است، زیرا برنامه هاي پایتون به وسیلهٔ مفسر اجرامی شوند.
- دو راه براي استفاده از مفسر وجود دارد: حالت خط فرمان و حالت اسکریپت.
- به Python Prompt، <<< گفته می‌شود که نشان می‌دهد مترجم برای دریافت دستورات آماده است.
- روش دوم: شما می توانید برنامه را در یک فایل بنویسید و از مفسر براي اجراي محتویات فایل استفاده کنید. چنین فایلی را اسکریپت می نامند (فایل هاي پایتون پسوند py. دارند).

نوشتن اولین برنامه

- نصب پایتون: Python download page
- بعد از اجرا

```
Python 3.5.1 (v3.5.1:37a07cee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
```

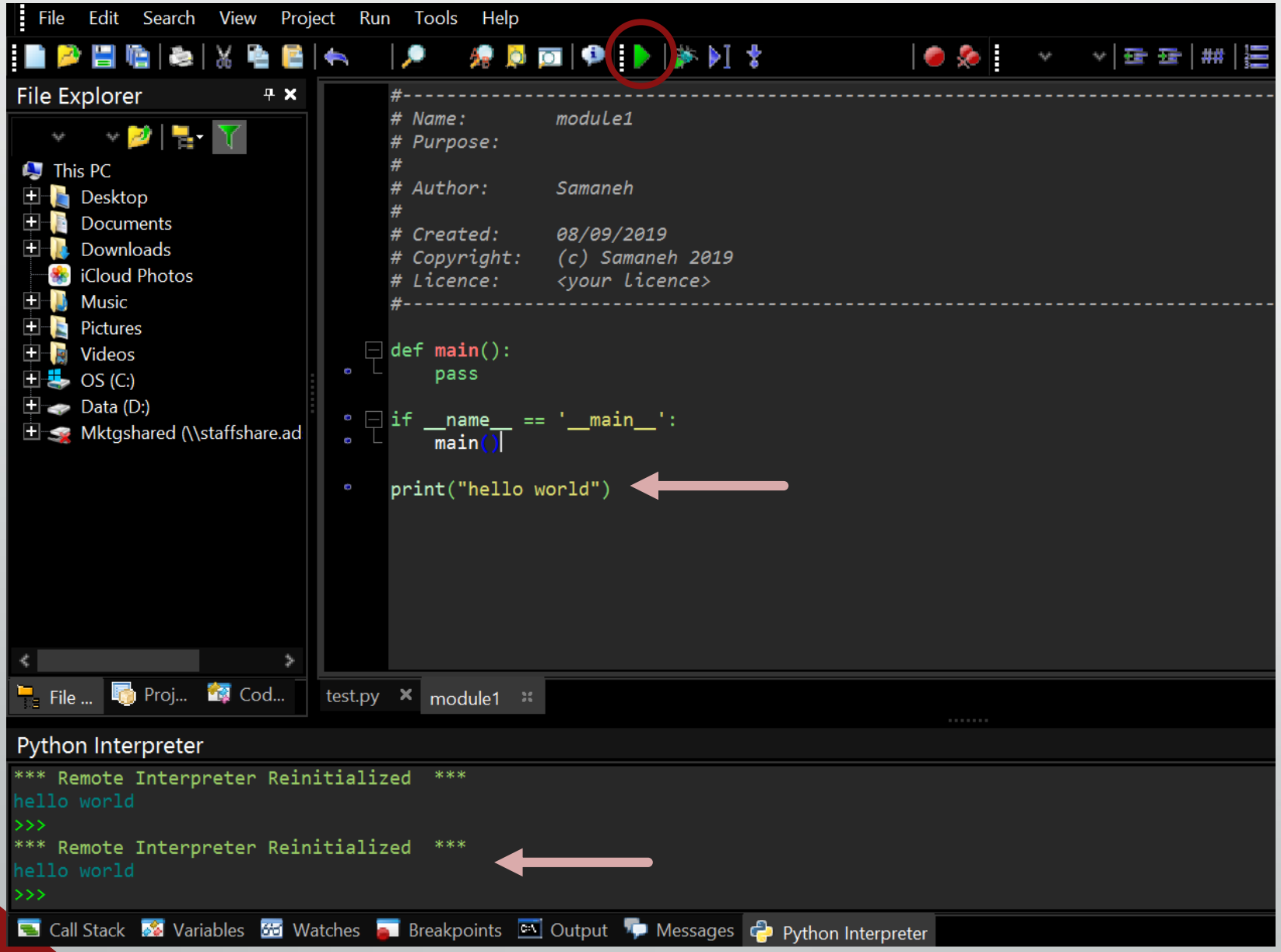
- کوتاه‌ترین برنامه به طور سنتی در برنامه‌سازی، برنامه‌ای است که یک عبارت را در صفحه نمایش چاپ کند

```
>>> print("Hello world")
Hello world
```

- روش دوم: نصب PyScripter IDE یا PyCharm IDE

در یک فایل خط زیر را بنویسید و اجرا کنید.

```
print("Hello World")
```


File Edit Search View Project Run Tools Help
File Explorer
This PC
Desktop
Documents
Downloads
iCloud Photos
Music
Pictures
Videos
OS (C:)
Data (D:)
Mktgshared (\\staffshare.ad
#----------
Name: module1
Purpose:
#
Author: Samaneh
#
Created: 08/09/2019
Copyright: (c) Samaneh 2019
Licence: <your licence>
#----------
def main():
pass
if __name__ == '__main__':
main()
print("hello world")
File ...
Proj...
Cod...
test.py
module1
Python Interpreter
*** Remote Interpreter Reinitialized ***
hello world
>>>
*** Remote Interpreter Reinitialized ***
hello world
>>>
Call Stack
Variables
Watches
Breakpoints
Output
Messages
Python Interpreter

ادامه مطلب در جلسه بعد

مقدمات برنامه‌سازی