



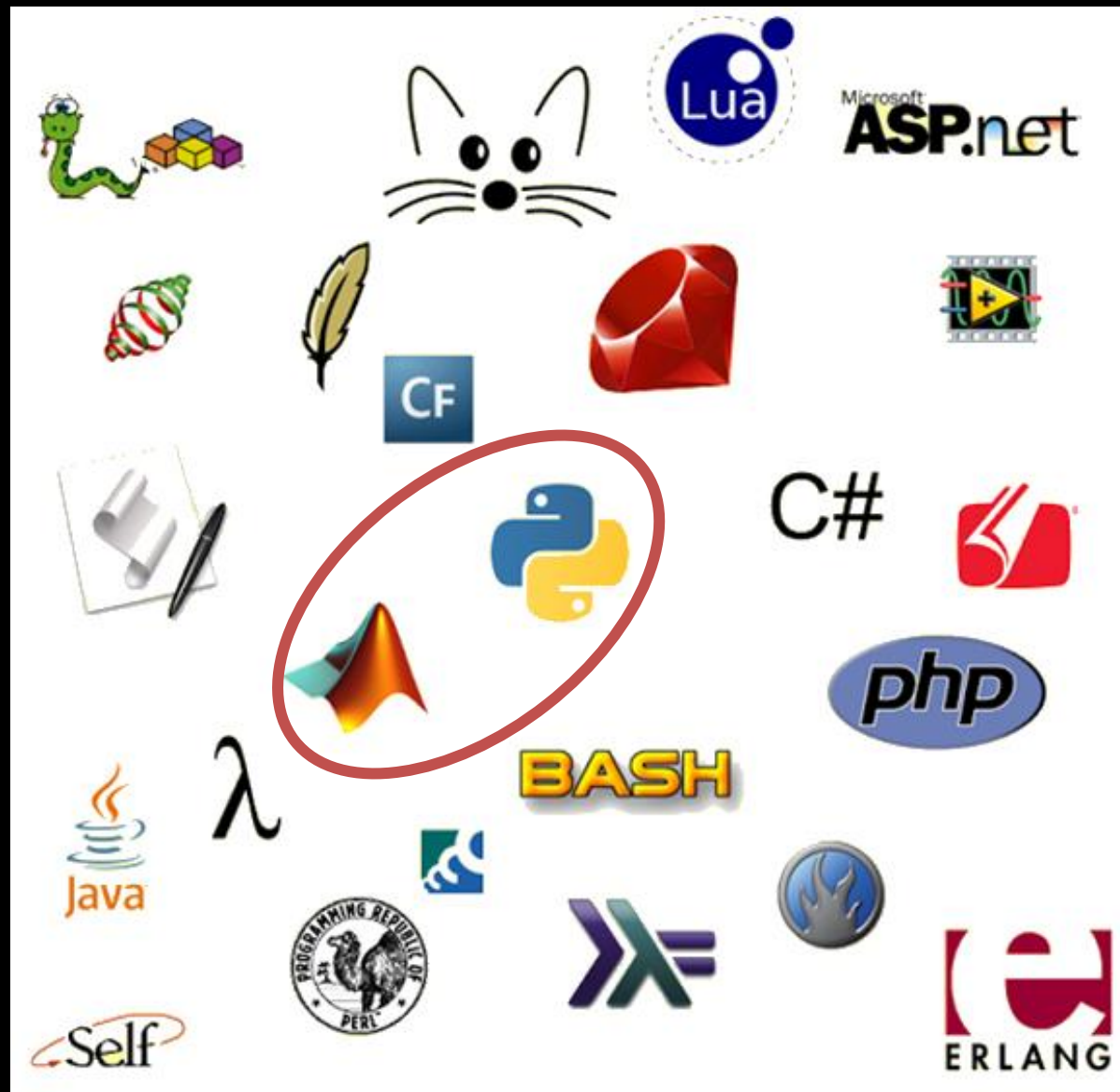
# HAB 619 Introduction to Scientific Computing in Sports Science



**SERDAR ARITAN**

[serdar.aritan@hacettepe.edu.tr](mailto:serdar.aritan@hacettepe.edu.tr)

Biyomekanik Araştırma Grubu  
[www.biomech.hacettepe.edu.tr](http://www.biomech.hacettepe.edu.tr)  
Spor Bilimleri Fakültesi  
[www.sbt.hacettepe.edu.tr](http://www.sbt.hacettepe.edu.tr)  
Hacettepe Üniversitesi, Ankara, Türkiye  
[www.hacettepe.edu.tr](http://www.hacettepe.edu.tr)



## What is numerical computations?

Mathematical algorithms, though usually invisible, are all around us. The microcomputer in your car controlling the fuel ignition uses a control algorithm embodying mathematical theories of dynamical systems; a Web search engine might use large-scale matrix computations; a “smart map” using a Global Positioning System to tell where you are and the best way to get home embodies numerous numerical and non-numerical algorithms. Behind these applications is software that does numerical computations.



## Scientific software language

In every decade since the 1950s, the complexity of scientific software has increased a great deal. Object-oriented software has come to the fore in scientific and engineering software with the development of a plethora of object-oriented matrix libraries and finite element packages. Fortran used to be the clear language of choice for scientific software. That has changed. Much scientific software is now written in C, C++, Java, **Matlab**, **Python**, and languages other than Fortran.

## Quantitative problems

**Numerical software** is the software used to do computations with real numbers; that is, with numbers with decimal points in them like  $\pi = 3.141\ 5926 \dots$ . These kinds computations are commonly of great **scientific** and engineering importance. Real numbers can be used to represent physical quantities (position, height, force, stress, viscosity, density, etc.). Computation with real numbers can be for simulating the human movement, finding the stresses in a long bone, or for determining how many muscle fiber unit can touch each other without penetrating.



## We want to go where no one has gone before!

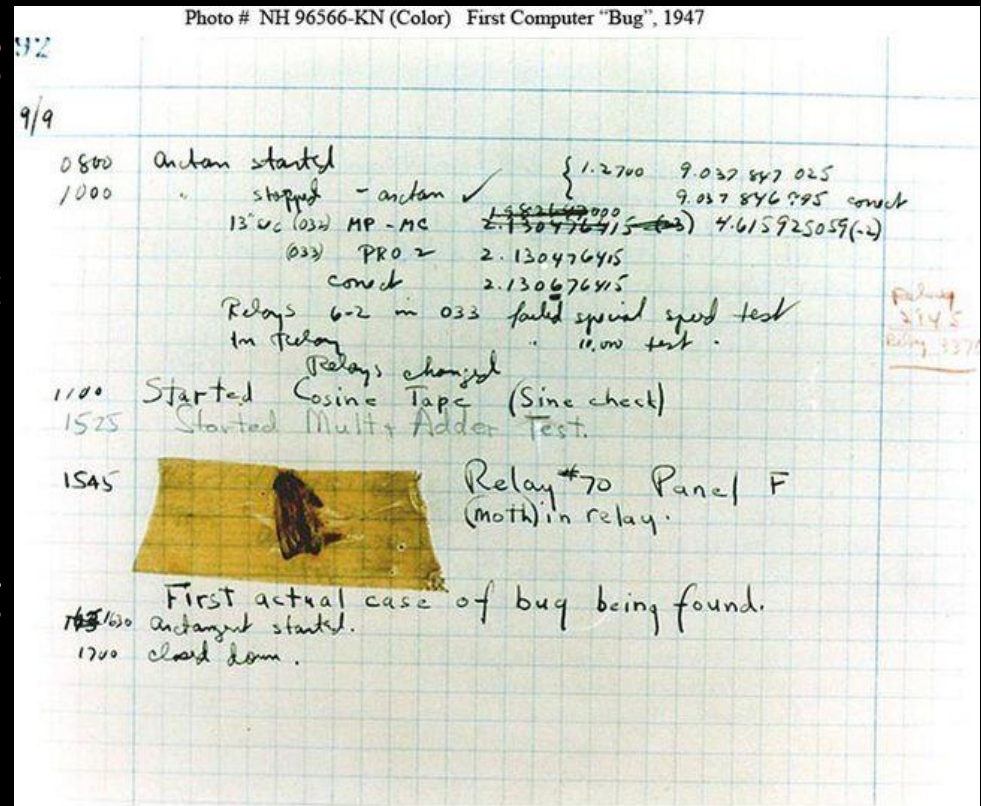
As scientists, we are interested in research. That means that we want to **go where no one has gone before**. It means that we want to **investigate problems** and approaches no-one else has thought of. We are **unlikely to get something profoundly** important on our first try. We will try something, see what happens, and then ask some new questions, and try to answer those. This means that **our software is going to have to change** as we have different problems to solve, and want to answer different questions. Our software will have to change quickly.

## Rapid prototyping

Rapidly changing software is a challenge. Each change to a piece of software has the chance to introduce **bugs**. A **bug** is a flaw or glitch in a system. Every time we change an assumption about what we are computing, we have an even bigger challenge to modify our software, since it is easy to build in bad or restrictive assumptions into our software. The challenge of rapidly changing software is not unique to research, but it is particularly important here.

# Why this glitch is called Bug?

“First actual case of bug being found,” one of the team members wrote in the logbook. The team at Harvard University in Cambridge, Massachusetts, found that their computer, the Mark II, was delivering consistent errors.





**The trapped insect had disrupted the electronics of the Harvard Mark II computer.**



# The Advantages of Python for Technical Programming

Python is supported on many different computer systems, providing large measure of independence. **Program written on any platform will run on all of the other platforms** and data files written on any platform can be read transparently on any other platform.

## ➡ Windows Specific

XP, 2000, NT 4.0 [Pentium III, IV, Xeon, Pentium M, AMD Athlon, Athlon XP, Athlon MP]

## ➡ UNIX/Linux Specific - 32-bit & 64-bit Operating Environment

Sun Solaris [SPARC ULTRA]

HP-UX [PA-RISC 2.0]

Linux [Pentium III, IV AMD Opteron, AMD Athlon, Athlon XP, Athlon MP]

Linux [AMD64 –AMD Opteron, AMD Athlon 64, Intel EM64T]

## ➡ Macintosh Specific

Mac OS X (*Panther*) [PowerMac G4, PowerMac G5]



## What is Python?



Snake logos and mascot not with standing, it's named after *Monty Python's Flying Circus*

*Monty Python* (sometimes known as The Pythons) was a British surreal comedy group that created Monty Python's Flying Circus, a British television comedy sketch show that first aired on the BBC on 5 October 1969.

# What is Python?

- Invented in the Netherlands, early 90s by Guido van Rossum
- Named after Monty Python
- Open sourced from the beginning, man-aged by [Python Software Foundation](https://python.org)
- Considered a scripting language, but is much more
- Scalable, object oriented and functional from the beginning
- Used by Google from the beginning



## What is Python?

“Python is an experiment in how much freedom program-mers need. Too much freedom and nobody can read another's code; too little and expressive-ness is endangered.”

- [Guido van Rossum](#)





# Python's Market Place

- TIOBE has been collecting data on programming language “popularity” for many years

Year	Winner
2022	C++
2021	Python
2020	Python
2019	C
2018	Python
2017	C
2016	Go
2015	Java
2014	JavaScript
2013	Transact-SQL
2012	Objective-C
2011	Objective-C
2010	Python
2009	Go
2008	C
2007	Python
2006	Ruby
2005	Java
2004	PHP
2003	C++

Sep 2023	Sep 2022	Change	Programming Language	Ratings	Change
1	1		Python	14.16%	-1.58%
2	2		C	11.27%	-2.70%
3	4	▲	C++	10.65%	+0.90%
4	3	▼	Java	9.49%	-2.23%
5	5		C#	7.31%	+2.42%
6	7	▲	JavaScript	3.30%	+0.48%
7	6	▼	Visual Basic	2.22%	-2.18%
8	10	▲	PHP	1.55%	-0.13%
9	8	▼	Assembly language	1.53%	-0.96%
10	9	▼	SQL	1.44%	-0.57%
11	15	▲	Fortran	1.28%	+0.26%
12	12		Go	1.19%	+0.03%
13	14	▲	MATLAB	1.19%	+0.13%
14	22	▲	Scratch	1.08%	+0.51%
15	13	▼	Delphi/Object Pascal	1.02%	-0.07%



# Python's Market Place

## TIOBE Index for Python

Source: [www.tiobe.com](http://www.tiobe.com)





## Distinct Features of Python

- Extensible (packages)
- Embeddable into applications
- Functional programming
- Object-Oriented programming
- Rapid Prototyping
- Great for readability and presentation
- White space is significant
- Low maintenance costs
- Exception handling
- Free (open source)



The core philosophy of the language is summarized by the document "PEP 20 (The Zen of Python)",

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Readability counts.

```
Try;  
>>>import this
```

# Download & Install winpython



The screenshot shows the WinPython website. At the top, there are navigation links: "releases", "overview", and "portable". Below these is the WinPython logo, which consists of a 2x2 grid of squares (blue, yellow, yellow, blue) and the text "WinPython". Below the logo is the tagline: "The easiest way to run Python, Spyder with SciPy and friends out of the box on any Windows PC, without installing anything!". The main content area is highlighted with a yellow border and contains the following text:

Project Home is on [Github](#), downloads pages are on [Sourceforge](#) and [Github](#), [md5-sha](#) , [Discussion Group](#)

## Recent Releases

Release [2023-03](#) of September 28th, 2023

Highlights (\*): Python-3.11.5, Jupyterlab-4.0.6, Numba-1.25, pandas-2.1.1, scipy-1.11.1, scikit\_learn-1.3.1, Poetry

WinPython **3.11** Downloads (\*\*) via [SourceForge](#) and [Github](#)

- WinPython64-**3.11.5.0**dot = Python 3.11.4 64bit only : [Changelog](#), [Packages](#)
- WinPython64-**3.11.5.0** = Python 3.11.4 64bit with PyQt5 + Spyder + Torch : [Changelog](#), [Packages](#)
- WinPython64-**3.11.5.0**mkl = Python 3.11.4 64bit with PyQt5 + Spyder + Mkl : [Changelog](#), [Packages](#)

Release [2023-02](#) of July 15th, 2023

Highlights (\*): Python-3.11.4, pandas-2.0.2, SQLAlchemy-2.0.15, Jupyterlab-3.6.5, opencv\_python, qrcode, python\_barcode

WinPython **3.11** Downloads (\*\*) via [SourceForge](#) and [Github](#)

<http://winpython.github.io/>





# Download & Install Anaconda

## Anaconda Installers

Windows 

Python 3.8

64-Bit Graphical Installer (457 MB)


32-Bit Graphical Installer (403 MB)

MacOS 

Python 3.8

64-Bit Graphical Installer (435 MB)

64-Bit Command Line Installer (428 MB)

Linux 

Python 3.8

64-Bit (x86) Installer (529 MB)

64-Bit (Power8 and Power9) Installer (279 MB)

<https://www.anaconda.com/products/individual>



# Numbers

Binary	Decimal	Hexadecimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

## The IEEE floating point system

The length of the mantissa describes how accurately numbers can be approximated.

Since

$$x \times b^e = (x_0 x_1 x_2 \cdots x_m \text{ base } b) \times b^e$$

$$= (x_0 + x_1 b^{-1} + x_2 b^{-2} + \cdots + x_{m-1} b^{-m+1}) \times b^e$$

The best known and most used floating point system is the IEEE floating point system, which is sometimes referred to as the IEEE 754 standard.

## The IEEE floating point system

IEEE 754 standard specifies a set of three different floating point formats: *single-precision*, *double-precision*, and *extended precision*.

A fundamental quantity for any floating point system is the unit roundoff. This is denoted by  $u$  and is the smallest positive number where the computed value of  $1 + u$  is different from 1. The **machine epsilon** is the smallest  $a - 1$  where  $a$  is the smallest representable number greater than 1.

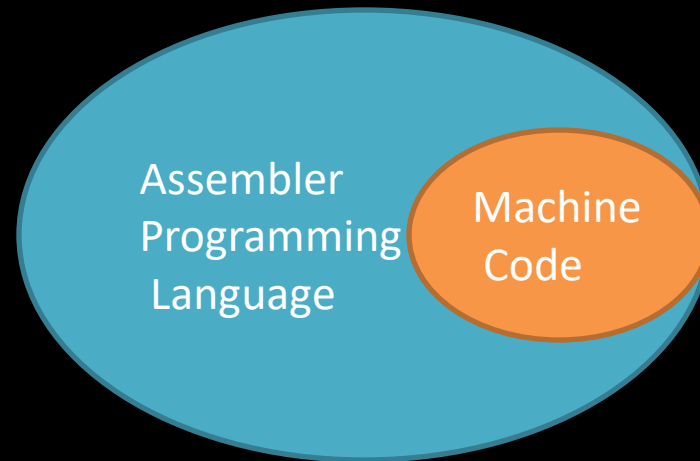
## The trouble with real numbers

Real numbers like  $\pi = 3.141\ 592\ 65 \dots$  are a problem for computers because (in general) they have an infinite number of digits. Unless your computer has infinite memory (and ours don't), there is no way it can store  $\pi$  exactly. So we do the next best thing, and store a reasonable approximation. Real numbers are stored as **floating point numbers**. Floating point numbers are numbers like  $\pi \approx 3.141\ 592\ 6$ , or Avogadro's number  $\approx 6.03 \times 10^{23}$ . These have the form  $\pm x \times b^e$  where  $x$ , the **significand** or **mantissa**,





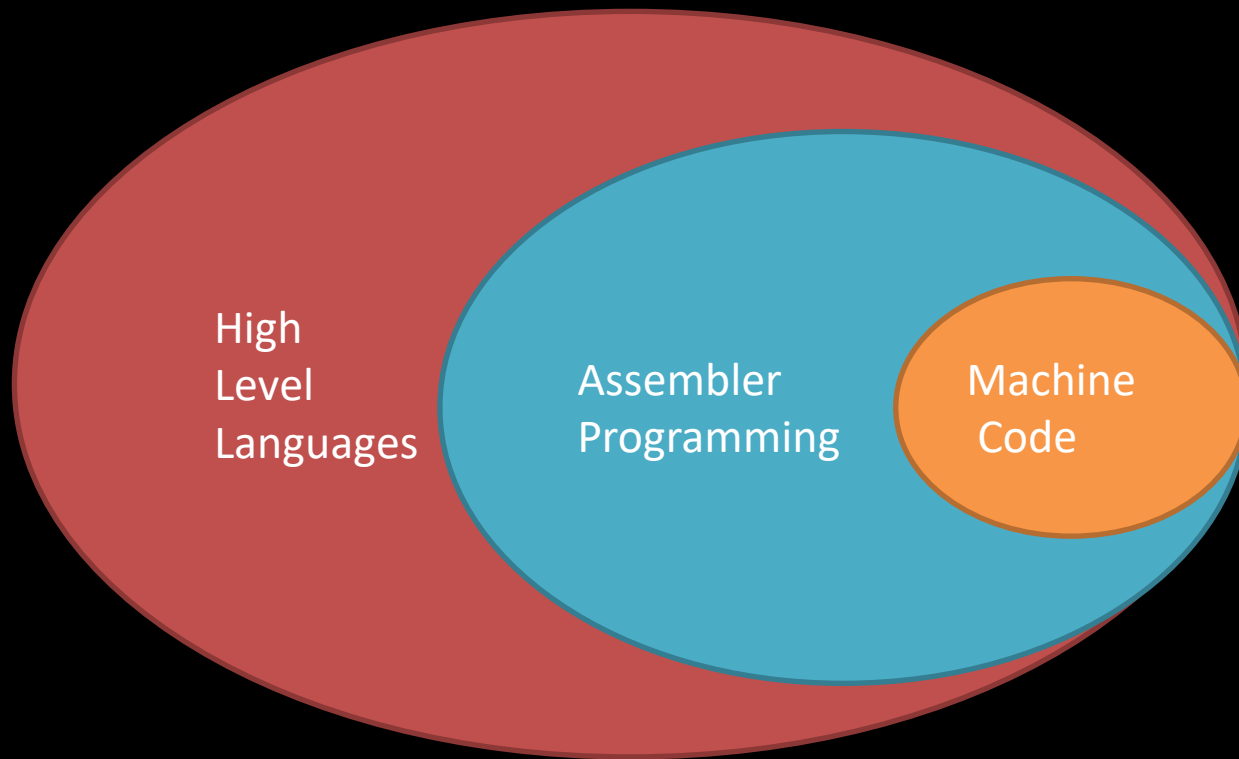
# Computer Programming



(1951-1959)



# Computer Programming



## Compiling and Interpreting

Both types of languages have their strengths and weaknesses. Usually, the decision to use an interpreted language is based on time restrictions on development or for ease of future changes to the program.

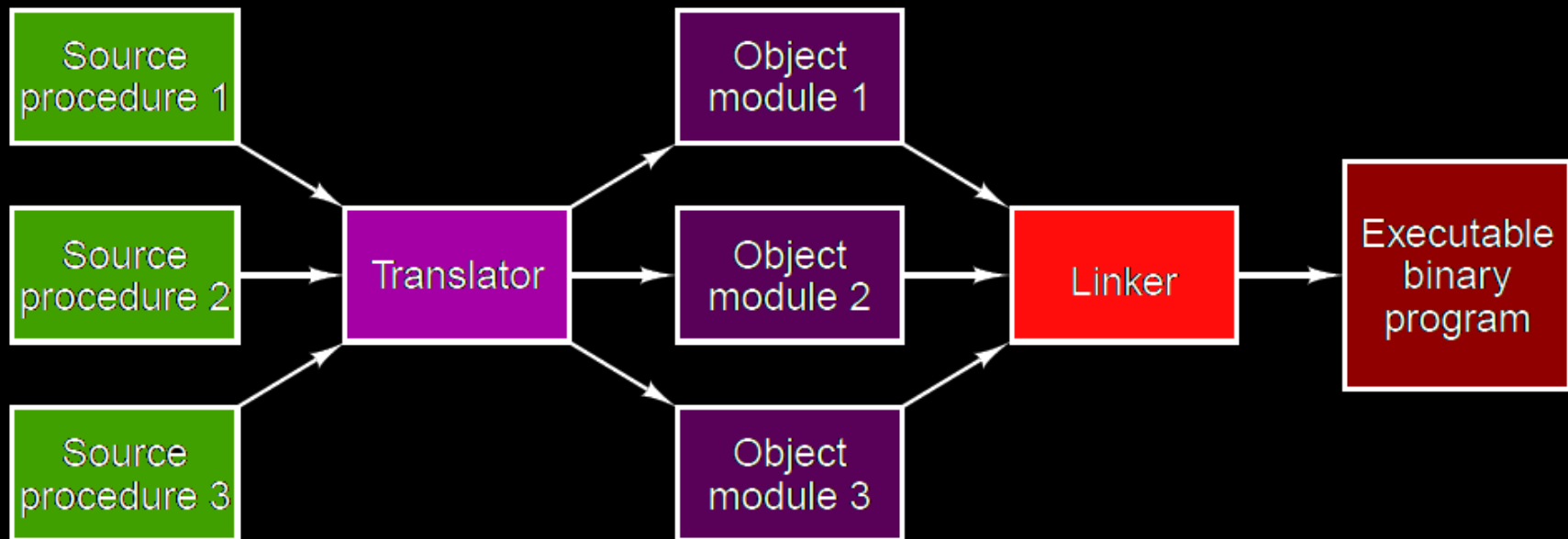
Compiled languages are all translated by running the source code through a compiler. This results in very efficient code that can be executed any number of times. The overhead for the translation is incurred just once, when the source is compiled; thereafter, it need only be loaded and executed. Interpreted languages, in contrast, must be parsed, interpreted, and executed each time the program is run, thereby greatly adding to the cost of running the program. For this reason, interpreted programs are usually less efficient than compiled programs.

## Compiling and Interpreting

An interpreter is a translating program that translates and executes the statements in sequence. Unlike an assembler or compiler that produces machine code as output, which is then executed in a separate step, an interpreter translates a statement and then immediately executes the statement.

By definition, machine code differs from machine to machine. That is, each type of CPU has its own machine language that it understands. So how can we give each of you the experience of using machine language when you may be working on different machines? We solve that problem by using a virtual computer. A virtual computer is a hypothetical machine, in this case one that is designed to contain the important features of real computers that we want to illustrate.

## Compiling and Interpreting

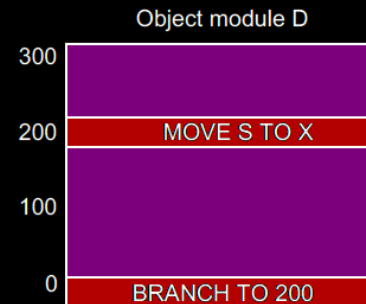
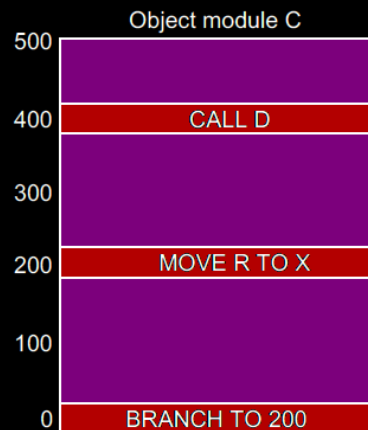
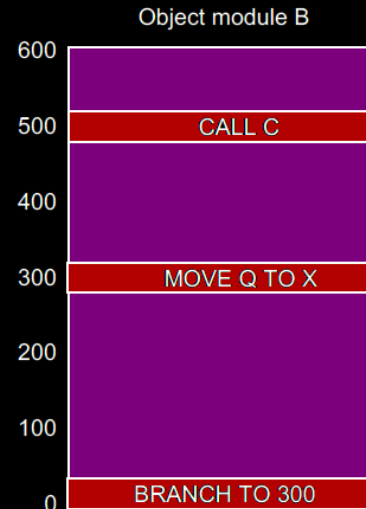
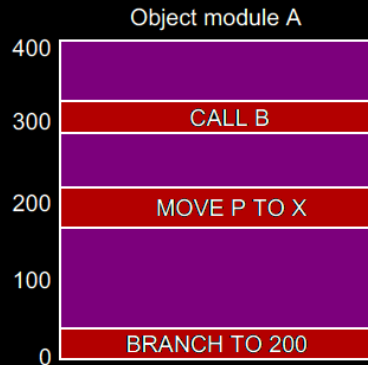


Generation of an executable binary program from a collection of independently translated source procedures requires using a linker.





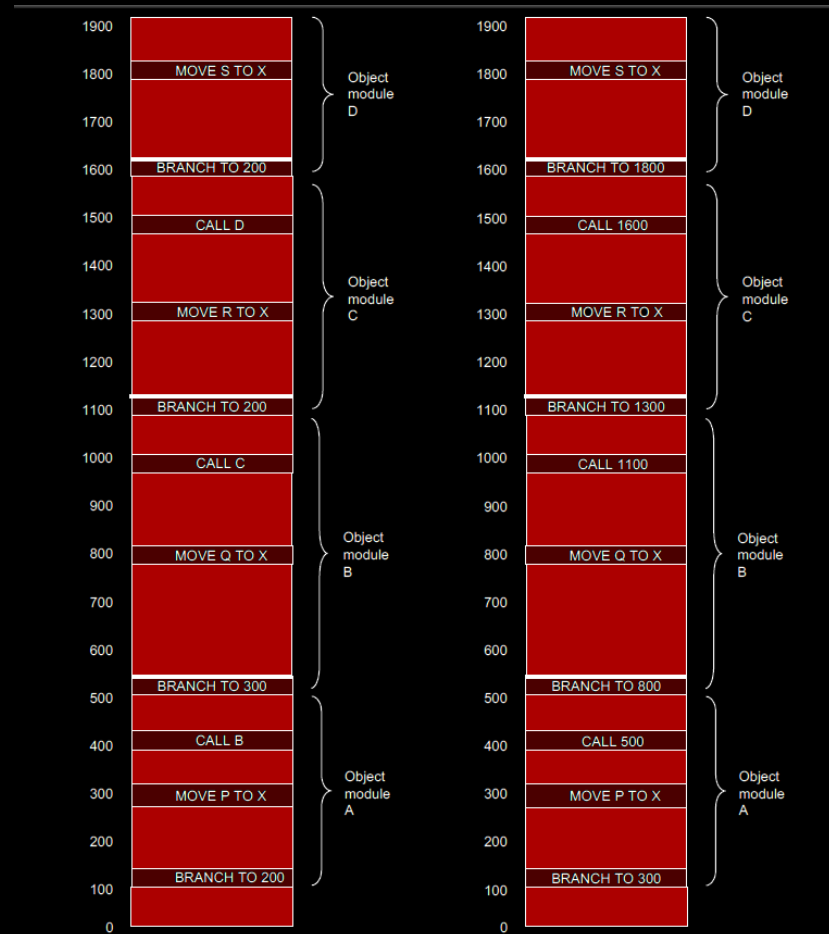
## Linker



Each module has its own address space, starting at 0.

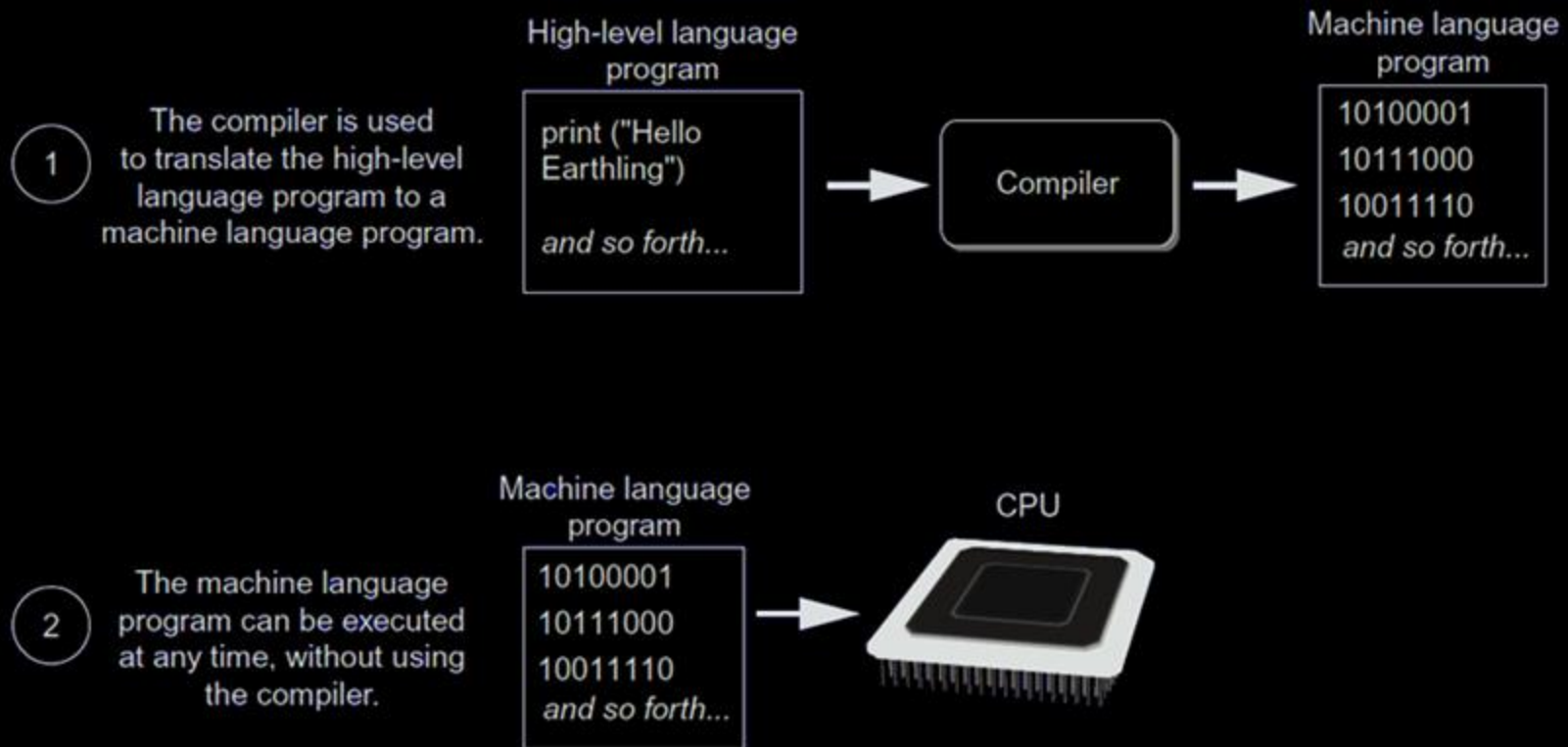


## Linker

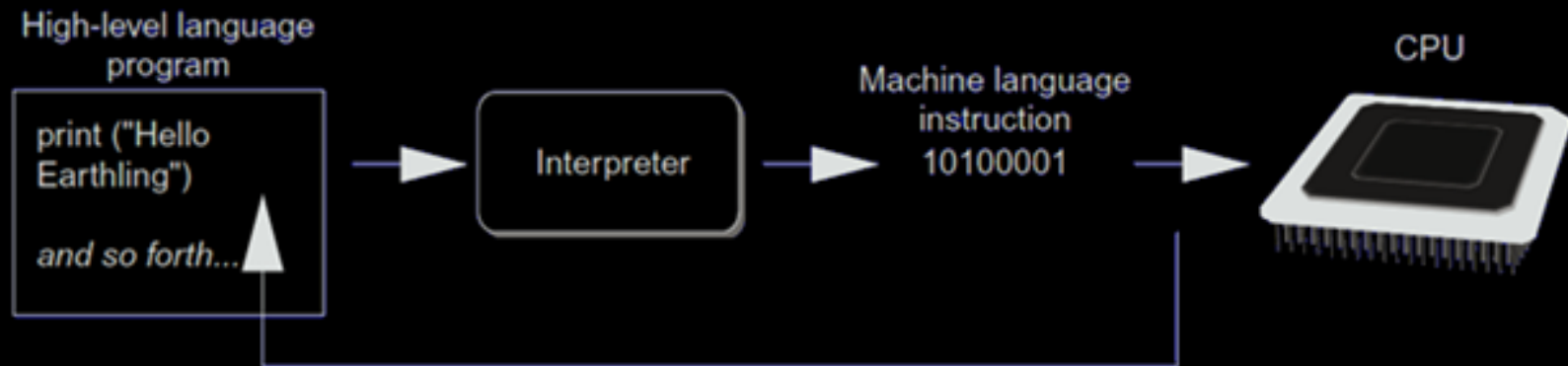


(a) The object modules of Fig. 7-14 after being positioned in the binary image but before being relocated and linked. (b) The same object modules after linking and after relocation has been performed. Together they form an executable binary program, ready to run.

## Compiling and Interpreting



## Compiling and Interpreting



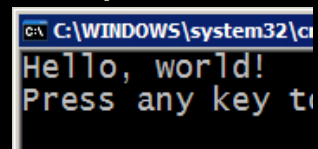
The interpreter translates each high-level instruction to its equivalent machine language instructions and immediately executes them.

*interpret*

source code  
Hello.py



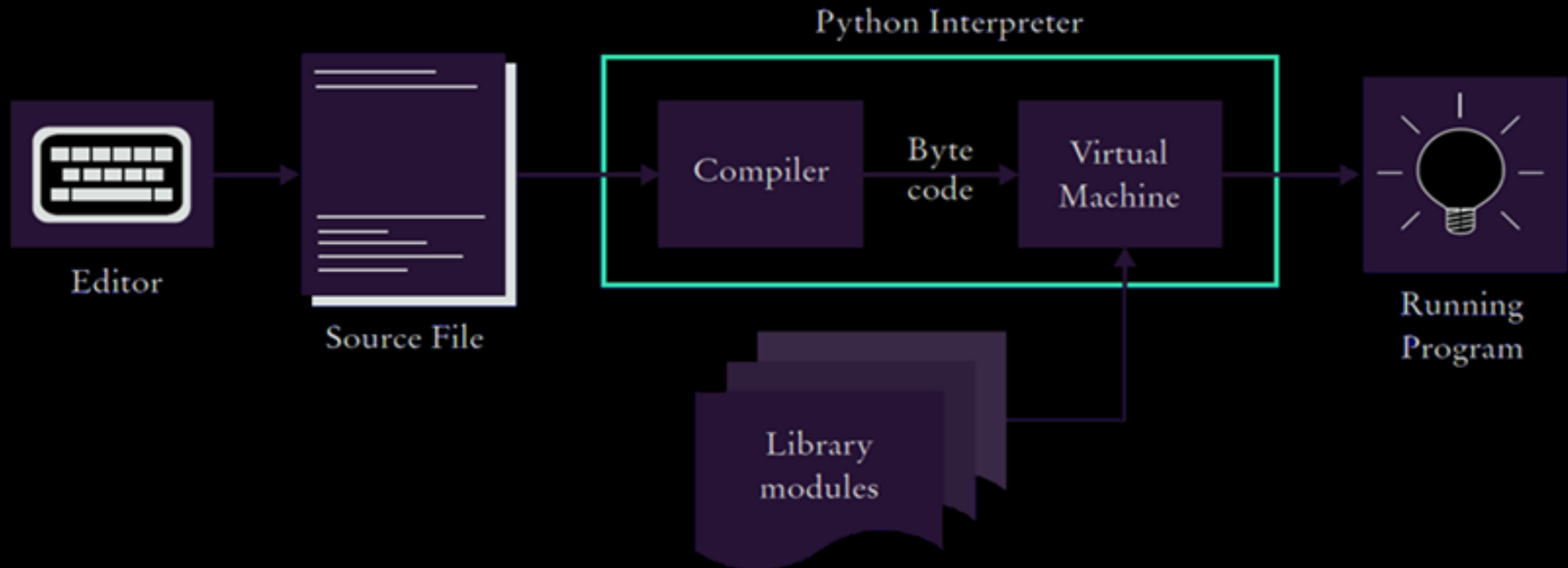
output



```
C:\WINDOWS\system32\cmd
Hello, world!
Press any key to continue
```

A screenshot of a Windows command prompt window showing the output of the Python program: `Hello, world!` and a prompt to press any key to continue.

## Compiling and Interpreting







# Compiling and Interpreting

The screenshot displays the Visual Studio IDE with a C program named 'HelloWorld.c' open. The code is as follows:

```
1 #include <stdio.h>
2
3 void main() {
4     printf("Hello, world.\n");
5 }
6
7
```

The 'Microsoft Visual Studio Debug Console' window shows the output of the program:

```
Hello, world.
C:\Development\Cpp\HelloWorld\Release\HelloWorld.exe (process 7948) exited with code 0.
Press any key to close this window . . .
```

The 'Output' window at the bottom shows the build process:

```
1>HelloWorld.c
1>Generating code
1>Previous IPDB not found, fall back to full compilation.
1>All 4 functions were compiled because no usable IPDB/IOB from previous compilation was found.
1>Finished generating code
1>HelloWorld.vcxproj -> C:\Development\Cpp\HelloWorld\Release\HelloWorld.exe
***** Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped *****
```

A status bar at the bottom indicates 'Build succeeded'.



# Compiling and Interpreting

```
Developer Command Prompt for VS 2019

*****
** Visual Studio 2019 Developer Command Prompt v16.8.4
** Copyright (c) 2020 Microsoft Corporation
*****

C:\Program Files (x86)\Microsoft Visual Studio\2019\Enterprise>cl
Microsoft (R) C/C++ Optimizing Compiler Version 19.28.29336 for x86
Copyright (C) Microsoft Corporation. All rights reserved.

usage: cl [ option... ] filename... [ /link linkoption... ]

C:\Program Files (x86)\Microsoft Visual Studio\2019\Enterprise>_
```

## Future of Computer!

- I think there is a world market for maybe five computers.

Thomas Watson, chair of IBM, 1943.

- Where . . . the ENIAC is equipped with 18,000 vacuum tubes and weighs 30 tons, computers in the future may have only 1,000 vacuum tubes and weigh only 1.5 tons.

*Popular Mechanics, 1949.*

- There is no reason anyone would want a computer in their home.

Ken Olson, president, chairman, and founder of Digital Equipment Corp., 1977.

- I predict the Internet . . . will go spectacularly supernova and in 1996 catastrophically collapse.

Bob Metcalfe, 3Com founder and inventor, 1995.



## Lecture Notes

Google serdar aritan

About 23,800 results (0.36 seconds)

**Serdar Aritan**  
Researcher

**Dr.Öğr.Üyesi SERDAR ARITAN | AVESİS**  
Diğer E-posta: [serdar.aritan@hacettepe.edu.tr](mailto:serdar.aritan@hacettepe.edu.tr); Web Sayfası: [yunus.hacettepe.edu.tr/~serdar.aritan](http://yunus.hacettepe.edu.tr/~serdar.aritan); İş Telefonu: +90 312 780 6893; İş Telefonu: +90 312...

**Serdar Aritan**  
Faculty Member, Hacettepe University - 598 tarafından alıntlandı - Biomechanics - Sports  
Biomechanics - Simulation - Computer Graphics

**About**  
**Education:** [Manchester Metropolitan University](#) (1994–1998), [MORE](#)  
[Claim this knowledge panel](#) [Feedback](#)

**Profiles**  
 [LinkedIn](#)



## Lecture Notes

### CONTACT

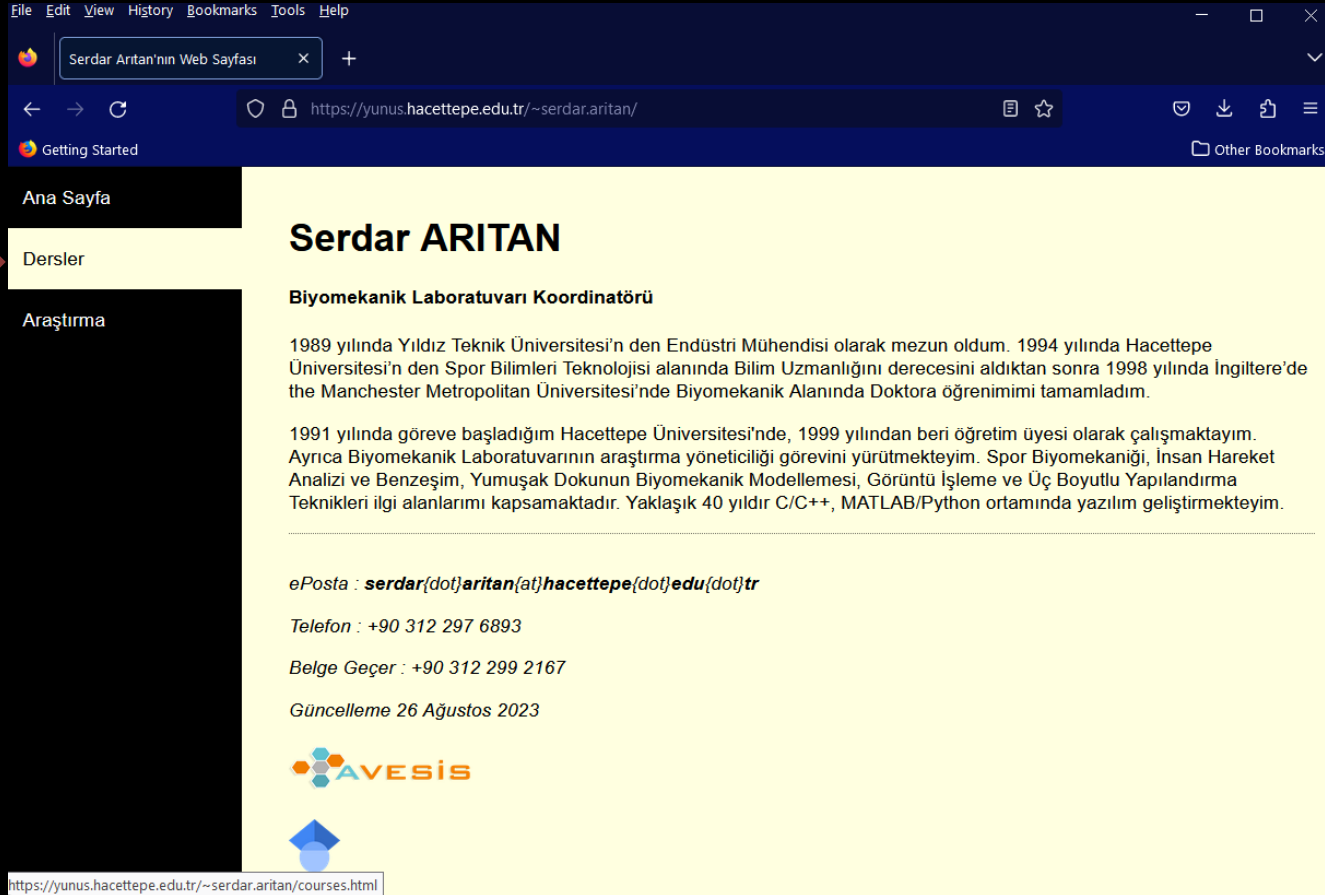


Other Email	serdar.aritan@hacettepe.edu.tr
Web Page	<a href="http://yunus.hacettepe.edu.tr/~serdar.aritan">yunus.hacettepe.edu.tr/~serdar.aritan</a>
Office Phone	+90 312 780 6893
Office Phone	+90 312 297 6893
Fax Phone	+90 312 299 2167
Office	Spor Bilimleri Fakültesi Oda No:2-8
Address	Hacettepe Üniversitesi Spor Bilimleri Fakültesi Biyomekanik Araştırma Grubu 06810 Beytepe Ankara





# Lecture Notes



File Edit View History Bookmarks Tools Help

Serdar Aritan'nın Web Sayfası x +

https://yunus.hacettepe.edu.tr/~serdar.aritan/

Getting Started Other Bookmarks

Ana Sayfa

Dersler

Araştırma

## Serdar ARITAN

**Biyomekanik Laboratuvarı Koordinatörü**

1989 yılında Yıldız Teknik Üniversitesi'n den Endüstri Mühendisi olarak mezun oldum. 1994 yılında Hacettepe Üniversitesi'n den Spor Bilimleri Teknolojisi alanında Bilim Uzmanlığını derecesini aldıktan sonra 1998 yılında İngiltere'de the Manchester Metropolitan Üniversitesi'nde Biyomekanik Alanında Doktora öğrenimimi tamamladım.


1991 yılında göreve başladığım Hacettepe Üniversitesi'nde, 1999 yılından beri öğretim üyesi olarak çalışmaktayım. Ayrıca Biyomekanik Laboratuvarının araştırma yöneticiliği görevini yürütmekteyim. Spor Biyomekanik, İnsan Hareket Analizi ve Benzeşim, Yumuşak Dokunun Biyomekanik Modellemesi, Görüntü İşleme ve Üç Boyutlu Yapılandırma Teknikleri ilgi alanlarımı kapsamaktadır. Yaklaşık 40 yıldır C/C++, MATLAB/Python ortamında yazılım geliştirmekteyim.


ePosta : **serdar{dot}aritan{at}hacettepe{dot}edu{dot}tr**

Telefon : +90 312 297 6893

Belge Geçer : +90 312 299 2167

Güncelleme 26 Ağustos 2023





https://yunus.hacettepe.edu.tr/~serdar.aritan/courses.html



# Lecture Notes

File Edit View History Bookmarks Tools Help

Serdar Arıtan'nın Yürütülen Dersler x +

← → ↻ https://yunus.hacettepe.edu.tr/~serdar.aritan/courses.html ☆

Getting Started Other Bookmarks

Ana Sayfa  
Dersler  
Araştırma

## Serdar ARITAN

### 2023-2024 Güz Yarıyılı (2 Ekim 2023 / 7 Ocak 2024)

- Bilişim Enstitüsü Lisansüstü Dersler

[BCO 601](#) Python Programlama

[BCO 602](#) Animasyon İçin Betik Diller

Sağlık Bilimleri Enstitüsü Lisansüstü Dersler

[HAB 619](#) Spor Bilimlerinde Bilimsel Programlamaya Giriş

Genel sınavlar (8 Ocak 2024 / 21 Ocak 2024)

- Lisans Dersleri

[ANR 305-02](#) Maç ve Yarışma Analizi [Şube 2 Bireysel Sporlar için Hareket Analizi]

[ANR 413](#) Bilgisayar Programlama

Genel sınavlar (8 Ocak 2024 / 21 Ocak 2024)

Bütünleme sınavları (29 Ocak 2024 / 4 Şubat 2024)

Ders Takvimi  
Salı, 29 Ağustos