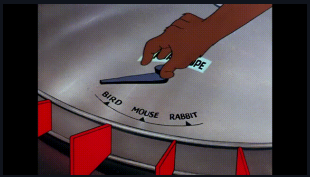
Object Oriented Programming

Mitsiu Alejandro Carreño Sarabia



Agenda

- Slides intro
- Course intro
- Generalities
- Tests
- Homework

Slides intro

Slides intro

Code snippets will be shown with highlight and execution:

```
1 def fib(n):    # Function definition
2     """Print a Fibonacci series less than n."""
3     a, b = 0, 1
4     while a < n:
5         print(a, end=' ')
6         a, b = b, a+b
7     print()
8
9 fib(2000)   # Call fib function with parameter 2000
```

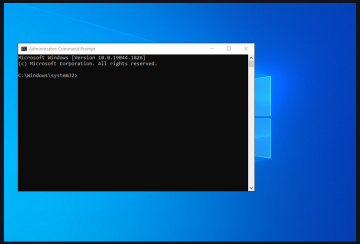
---------- [finished] ----------

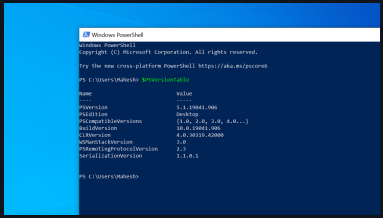
```
0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597
```

Slides intro

At the bottom of my screen you'll see two tabs "Slides" & "Terminal" the active tab is marked in green

The terminal window is a powerfull tool which will allow us to execute commands and programs, you are required to learn how to use the terminal

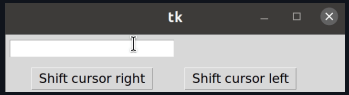




Powershell

I personally prefer text-based rather than graphical interfaces, it's faster!

- Mis-clicks
- Loose the cursor
- Time spent to move coursor



So I rarely use it.

You can use any editor or environment which suits you, I encourage you to learn your tools so you can set them up for your use and confort.

Slides intro

All this slides will be shared, as well as source code and extra material so here's my recommendation:

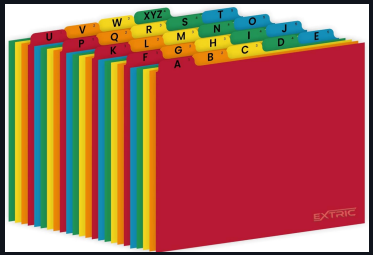
- Take hand written notes in a notebook
- Write down definitions
- Write down examples or analogies which makes sense to you
- Write down diagrams

Course intro

Course intro

- Why this career?
- Do you enjoy writting code?
- Which is the most complex program you've written?
- Which programming languages do you know? Favourite?
- Which editor/tools do you use to write code?
- Do you know what is git?
- Which Operating system do you use?
- Which Operating system did you use on yout OS class las quarter?
- What's your opinion on tech certificates?
- Do you enjoy reading? (Not limited to books)
- To whom do you write code to?

Object oriented programming is a programming model which uses objects, linked via functions, to solve problems. The main idea is simple: organize programs in the image and likeness of the organization of real objects in the world



Let's focus on organizing which is the best way to organize the material (source code, notes, slides, etc) you've archived during the career?

- By course

```
├── English I
├── OS
└── Calculus
```

- By period (four-month period)

```
├── Q1
│   └── English I
├── Q2
│   ├── OS
│   └── Calculus
└── Q3
    └── OOP
```

- By type (eg /Slides, /Homeworks /Source code)

```
├── Slides
│   ├── OS
│   └── Calculus
├── Homework
│   ├── English I
│   └── Calculus
└── SourceCode
    └── OOP
```

In my opinion there's no general best way, it depends on what's to optimize

- By course is best if we want to search but are unsure of the period
- By period is best if we want to search chronologically
- By type is best of we want to seach if we dont remember the course/period it belong to

In a similar fashion there are multiple way's to organize and structure our source code, each with it's own strengths and weaknesses

- Structured programming: Code execution is seen as sequential best fit for simple flows
- Object Oriented: Code is organized as objects each with a set of atributes and methods, best fit for multi-entity programs
- Functional: Code is organized into pure functions without state, best fit for crital and secure code execution
- Data Oriented: Code is organized based on physical locallity, best fit for execution performance

Course intro

Defitions:

Object Oriented Programming is a paradigm
Paradigm

1. Noun. A model of something, or a very clear and typical example of something

> A theory or set of theories whose central core is accepted without question and which provides the basis and model for solving problems and advancing knowledge.

Programming paradigm

1. A high level model to conceptualize and structure a computer program implementation

▓▓ Course intro

▓▓ Defitions:

Implementation

> The process of moving an idea from concept to reality

Are this "implementations" in the room with us right now?

YES! **Implemetation to get the Fibonacci series**

```
1 def fib(n):    # Function definition
2     """Print a Fibonacci series less
  than n."""
3     a, b = 0, 1
4     while a < n:
5         print(a, end=' ')
6         a, b = b, a+b
7     print()
8
9 fib(2000)   # Call fib function with
  parameter 2000
```

▒▒▒▒ Generalities

▒▒▒ Generalities

Requirements:

Mandatory:

- Computer
- Master your source code editor
- Master programming fundamentals (variables, if, for, functions)
- Commitment
- Computers ready to share (adapters)

Optional but recommended:

- Physical notebooks
- Get a lot of tokens
- Disable AI assistants (copilot, chatgpt, etc)
- Disable autocompletion in your editor

Generalities

Formal definitions have this format:

> Formal definition: Explains the concept at an academic level (descriptions found in books, articles, official documentation etc)

Formal definitions might be a little abstract or lack context explanations

So:

- In this slides you'll find informal definitions coloured red to explain the same concept in easier words

The ideal solution is for you to understand informal definitions as context for the formal definitions, use both to solve tests

Tests

Tests

Most tuesdays we'll have one of the following tests:

1. **Weekly test**

- Are solved by hand
- Have NO grade value
- Perfect score is traded for 2 tokens

2. **Partial test**

- 1st Partial = 30%
- 2nd Partial = 30%
- 3rd Partial = 40%

Partial test

- 1st Partial = 30%
  - Theorical evaluation = 50%
  - Practical evaluation (paper based, NO computer) = 50%
- 2nd Partial = 30%
  - Theorical evaluation = 40%
  - Practical evaluation (paper based, NO computer) = 60%
- 3rd Partial = 40%
  - Theorical evaluation = 40%
  - Practical evaluation (paper based, NO computer) = 60%

Homework

Homework

Master your code editor:

- Search in a single file
- Search in multiple files
- Know filename and file path of open file
- Go to definition
- Split screen
- Go to a specific line in a file
- Find and replace in a single file
- Find and replace in multiple files

Bring your computer next session

Upskill your english

Master how to keypress () [] {}

Practice PascalCase with shift key

References

References

PROGRAMACIÓN ORIENTADA A OBJETOS CON C++. Ceballos, 5th EDITION. (2018).
https://dle.rae.es/paradigma?m=form
https://dictionary.cambridge.org/dictionary/english/paradigm