Introducción al Pensamiento Computacional: Memorias sobre la Construcción de una Materia Necesaria

Franco Leonardo Bulgarelli, Gustavo Ernesto Trucco

1 Resumen **2**

2 Introducción **2**

2.1 Sobre el pensamiento computacional 2

2.2 Habilidades del pensamiento computacional 4

2.3 Antecedentes en el abordaje universitario del pensamiento computacional 4

3 Reseña histórica de la materia **6**

4 La materia hoy **6**

4.1 Programa 6

4.2 Carga horaria 7

4.3 Enfoque del dictado 8

4.3.1 Pensamiento computacional 8

4.3.2 Programación 9

4.3.3 Análisis de datos 11

4.4 Forma de evaluación 12

4.4.1 Trabajo práctico integrador de análisis de datos 12

4.4.2 Parcial 14

5 Trabajo futuro **15**

6 Conclusiones **16**

1 Resumen

En esta memoria describiremos la asignatura Introducción al Pensamiento Computacional (IPC, de ahora en más), materia joven gestada en la Universidad del CEMA, su proceso de construcción, su programa actual y sus horizontes futuros. Además, realizaremos una necesaria introducción al concepto de pensamiento computacional.

El objeto de este documento es doble: por un lado aportar posibles definiciones y aproximaciones entendiendo que el pensamiento computacional es un campo amplio y novedoso. Por el otro, buscamos acercar una perspectiva local y contextualizada al abordaje de esta disciplina que se sume a las múltiples experiencias pioneras que se pueden encontrar en otras universidades a lo largo del mundo.

2 Introducción

2.1 Sobre el pensamiento computacional

Antes de describir en detalle la asignatura IPC, es necesario desarrollar brevemente qué entendemos por pensamiento computacional quienes la dictamos.

En primer lugar, resulta fundamental reconocer que, dado que se trata de un campo jóven, no existe una definición única[1]. Existen definiciones por ejemplo que lo ubican como la fricción o límite de las interfaces humano-máquina, es decir, las cosas que necesitamos aprender para comunicarnos con una computadora, que no puedan ser abstraídas por una interfaz gráfica[2].

Por otro lado, una posible definición generalmente aceptada es que el pensamiento computacional describe los procesos mentales usados para formular problemas y expresar sus soluciones en términos que una computadora pueda utilizar de forma efectiva[3]. Sin embargo, existe un creciente interés en identificar los conceptos que involucra y su relación

[1] As computational thinking is a newborn field, its definition varies from researcher to researcher. Cansu, F. K., & Cansu, S. K. (2019). An Overview of Computational Thinking. International Journal of Computer Science Education in Schools, 3(1), 17-30. https://doi.org/10.21585/ijcses.v3i1.53

[2] A minimal definition of Computational Thinking: The stuff that we have to learn in order to communicate our tasks with a computer. It should be small, and the part that we learn should be generative, useful for new problems and new thinking. Everything else should be eliminated by good user interfaces. Guzdial, Mark (2019), A new definition of Computational Thinking: It's the Friction that we want to Minimize unless it's Generative, https://computinged.wordpress.com/2019/04/29/what-is-computational-thinking-its-the-friction-that-we-want-to-minimize/

[3] Thought processes used to formulate a problem and express its solution or solutions in terms a computer can apply effectively. Wing, Jeannette (2014). Computational Thinking Benefits Society. 40th Anniversary Blog of Social Issues in Computing, http://socialissues.cs.toronto.edu/index.html

entre ellos en lugar de brindar definiciones taxativas. Sumado a esto, diversos autores y organizaciones hacen particular foco en habilidades transversales y su aplicabilidad por fuera de la resolución de problemas empleando meramente una computadora[4].

Por lo tanto y teniendo en cuenta estas consideraciones, llegamos a la siguiente definición ampliada:

El pensamiento computacional es un conjunto de habilidades humanas que permiten formular problemas cuyas soluciones sean representables en términos de procesos o algoritmos. Dichas soluciones deben poder ser ejecutadas de forma precisa por un ente procesador de información, como una computadora o persona.

Como acotación final, es justo señalar que el término no está libre de críticas por sus inespecificidades, existiendo incluso importantes voces que invitan a entender al pensamiento computacional como un sinónimo de la enseñanza de las ciencias de la computación en el ámbito escolar[5]. Por eso es que, aún recuperando las visiones que abogan por la identificación de un campo con vuelo propio, al igual que otros autores entendemos conveniente no disociar el pensamiento computacional de las ciencias de la computación y la ingeniería de software[6].

En otras palabras, pese a que los conceptos de pensamiento computacional trascienden a la informática y son aplicables en contextos libres de computadoras, creemos que para una correcta estimulación de sus habilidades es esencial aprender a programar e incorporar saberes específicos de la ciencia de la computación.

Con un enfoque ecléctico y totalizador por igual, presentamos así la asignatura IPC, en la que conviven la práctica de la programación y la incorporación de conceptos fundamentales de la informática, la resolución de problemas lógicos cotidianos, y el pensamiento crítico sobre el mundo digital.

---

[4] *El Pensamiento Computacional es un concepto emergente que se entiende como una manera de pensar que no se restringe al código, la programación y la computadora. En este sentido, los estudiantes aprenden razonamiento lógico, pensamiento algorítmico y técnicas de resolución de problemas, todos conceptos y habilidades valiosas más allá del área específica de computación,* Plan Ceibal (2017) ¿Qué aporta al aula el Pensamiento Computacional?, https://www.ceibal.edu.uy/es/articulo/que-aporta-al-aula-el-pensamiento-computacional

[5] *Dado que el término de pensamiento computacional parece haber llegado para quedarse, invitamos a pensarlo como un sinónimo escolar de ciencias de la computación, que es una disciplina del conocimiento científico cuyo corpus de contenido está claramente definido. Esto permite tomar decisiones no arbitrarias al momento de diseñar programas de estudios para abordar estos temas.* BONELLO, M.B. Y SCHAPACHNIK, F. (2020). Diez preguntas frecuentes (y urgentes) sobre pensamiento computacional Virtualidad, Educación y Ciencia, 20 (11), pp. 156-167

[6] (...) we see computational thinking as being heavily informed by the fields of engineering and computer science (...) Weintrop, D., Beheshti, E., Horn, M. et al. Defining Computational Thinking for Mathematics and Science Classrooms. J Sci Educ Technol 25, 127–147 (2016). https://doi.org/10.1007/s10956-015-9581-5

2.2 Habilidades del pensamiento computacional

Si bien distintas fuentes hacen hincapié en diferentes habilidades del pensamiento computacional, las cuatro más comunes suelen ser:

- Descomposición en subproblemas.
- Identificación de patrones.
- Creación de abstracciones.
- Diseño de algoritmos.

Sin embargo, el concepto de abstracción es amplio y puede aludir a diferentes habilidades más específicas, como por ejemplo el proceso de simplificación o la construcción de representaciones[7], entre otras. Por este motivo y para evitar ambigüedad, reemplazamos a ésta por las nociones asociadas de representación de la información y la nominación de conceptos.

Por otro lado, consideramos importante explicitar el aprendizaje a través del error como habilidad fundamental del pensamiento computacional, dado que se trata de una herramienta frecuentemente aplicada en el campo[8].

Es por ello que el listado sobre el que trabajamos es el siguiente:

- Descomposición en subproblemas
- Identificación de patrones.
- Representación de la información.
- Nominación de conceptos.
- Diseño de algoritmos.
- Aprendizaje a través del error.

2.3 Antecedentes en el abordaje universitario del pensamiento computacional

[7] *Abstraction (as a process is) the process of reducing complexity by focusing on the main idea (...) (but as a product is) a new representation of a thing, a system, or a problem that helpfully reframes a problem by hiding details irrelevant to the question at hand.* K–12 Computer Science Framework (2016), p. 259. Retrieved from http://www.k12cs.org

[8] *(...) we argue for a pedagogical approach of embracing "mistakes" as an effective teaching tool for coding. This technique of teaching uses mistakes as a learning tool to teach computational thinking.* W. Scott Harrison and Nadine Hanebutte. 2018. Embracing coding mistakes to teach computational thinking. J. Comput. Sci. Coll. 33, 6 (June 2018), 52–62.

Previamente ya discutimos qué entendemos por pensamiento computacional y sus habilidades. ¿Pero cómo dar una introducción a éste en un contexto universitario, cuatrimestral, para un público diverso?

Nuevamente, nos hemos encontrado con decenas de experiencias diferentes, igualmente seminales, a lo largo del mundo. Podemos encontrar algunas como por ejemplo la del Doctor Hector Levesque, en la universidad de Toronto, que dicta un curso a estudiantes de disciplinas tan diversas como sociología, criminología, biología, cine y economía (por citar algunas) que gira en torno a la programación lógica usando el lenguaje de programación Prolog, que comenzó alternando, en sus propias palabras, "trabajo técnico con discusiones filosóficas", convencido de que llevar Python a un público tan heterogéneo sería muy difícil[9].

Por otro lado, la universidad de Pennsylvania ofrece a través de la plataforma Coursera[10] un curso de Pensamiento Computacional para la Resolución de Problemas, que toma la dirección contraria y su curso se basa en la presentación de algoritmos bajo un enfoque de programación imperativa con nociones de programación orientada a objetos, utilizando el lenguaje Python, y trabajando conceptos de arquitectura de computadoras y las habilidades de descomposición, reconocimiento de patrones, representación de información, abstracción y diseño de algoritmos[11].

En propuestas aún más sincréticas, encontramos la Universidad de Bologna, que sirve como Introducción a los Algoritmos, las Estructuras de Datos, y la Sintaxis y Semántica de Lenguajes (combinando varias de las asignaturas que podríamos encontrar en los primeros años de una carrera de Ciencias de la Computación o Ingeniería Informática), al tiempo que presenta lecturas sobre autores clásicos de las ciencias de la computación como Noam Chomsky, Grace Hooper, Dondald Knuth y de la literatura Latinoamericana, como Jorge Luis Borges y Gabriel García Marquez[12].

Como último ejemplo de las diferencias de interpretación en el ámbito educativo, podemos citar la Maynooth University, que ofrece un *BSc Computational Thinking* de 3 años que además de contar con 18 materias sobre Ciencias de la Computación (en las que se exploran lenguajes de diferentes paradigmas como C, R, Haskell, Java, SQL y Python), tiene otras 18 de matemática y 4 sobre filosofía[13].

---

[9] *The first-year students who take this course are not in a technical stream. I've had students majoring in sociology, criminology, commerce, history, biology, European studies, economics, political science,curatorial studies, cinema, English, and religion. I clearly would have a hard time teaching them to write interesting AI programs in Python or Scheme in a single year. However, I found that it was possible to teach nontechnical students to program in Prolog provided one did not insist on teaching them algorithms. It is possible, in other words, to get students to express what they need as a Prolog program and let Prolog search for answers. (…) At first I did a hybrid course, with a bit of technical work and a bit of philosophical discussion.* Levesque, Hector J. (2012) Thinking As Computation : A First Course

[10] https://es.coursera.org/

[11] https://www.coursera.org/learn/computational-thinking-problem-solving#syllabus

[12] https://www.unibo.it/en/teaching/course-unit-catalogue/course-unit/2019/424624

[13] https://www.maynoothuniversity.ie/study-maynooth/undergraduate-studies/courses/bsc-computational-thinking

Ante esta diversidad de contenidos en la propuesta podemos encontrar cuatro ideas fundamentales que son comunes a la mayoría de estas y otras experiencias:

1. La propuesta debe incluir una introducción práctica a la programación.
2. La propuesta debe tratar conceptos básicos de las ciencias de la computación, lógica y matemática.
3. El lenguaje Python es una herramienta que goza de gran aceptación en este tipo de propuestas.
4. Para que la propuesta mantenga la atención e interés, es deseable aplicar un enfoque multidisciplinar.

Estas ideas son, por tanto, elementos fundamentales del programa de IPC que describiremos en este documento.

3 Reseña histórica de la materia

A mediados de 2018 la Universidad del CEMA, atendiendo a las nuevas necesidades educativas que plantea el cada vez más omnipresente universo digital, decidió crear un curso piloto que buscaría acercar a estudiantes de sus carreras de Administración de Empresas, Relaciones Internacionales, Marketing y Economía al mundo de la informática. El planteo se apartaba por igual de cursos de ofimática como de cursos introductorios al desarrollo de software, buscando una propuesta superadora que fuera igualmente motivante para estudiantes con intereses diversos pero que fuera maś allá del mero uso de herramientas de productividad.

Surgió así la materia Introducción al Pensamiento Computacional, dictada en su primer curso por el Ingeniero Gustavo Brey, que dejó planteados cimientos sólidos para que el año siguiente esta experiencia piloto se convirtiera en asignatura obligatoria del primer año para las carreras antes mencionadas, y otras por venir (como por ejemplo, la Licenciatura en Negocios Digitales). En esta primera experiencia, el curso giró en torno a cuatro módulos: una presentación del contexto digital global, la resolución de problemas cotidianos, el aprendizaje de la programación utilizando los lenguajes Scratch[14] y Python, y una introducción a conceptos de análisis de datos.

En los cuatrimestres subsiguientes se amplió el equipo docente sumando a la Ingeniera Nadia Finzi, y a los Ingenieros Franco Bulgarelli y Gustavo Trucco (quienes escribimos estas líneas). Este nuevo equipo dictó 9 cursos completos para cerca de 400 estudiantes, y realizó modificaciones y profundizaciones que terminaron de definir la materia que se describe en esta memoria.

[14] https://scratch.mit.edu/

4 La materia hoy

4.1 Programa

La materia cuenta con tres ejes principales: el pensamiento computacional, la programación y el análisis de datos. El último, que se desarrolla en la segunda parte de la materia, tiene el rol de integrar los otros dos, y sirve también como una aplicación práctica y motivante de los mismos aplicando un enfoque de aprendizaje basado en proyectos (APB).

Estos ejes se abordan a lo largo de 5 unidades:

- **Unidad 1**: Introducción al pensamiento computacional. Sistemas informáticos. Concepto de programa. Lenguajes naturales y formales, lenguajes educativos e industriales. Gobstones. Aprendizaje automático (*machine learning*) y sus aplicaciones. Recopilación de datos. Sesgos de la información. Predicciones y cisnes negros. Dilemas éticos.
- **Unidad 2**: Entendimiento de Problemas y Patrones. Entender el quién, por qué y para qué. Divide y Conquistarás. Complejidad. Reconocimiento de Patrones. Procedimientos. Repetición
- **Unidad 3**: Abstracción, Simplificación y Generalización. Representación de información. Principios ontológicos. Niveles de abstracción. Detalles de Información para Modelado. Lenguajes textuales. Sintaxis
- **Unidad 4**: Diseño de Algoritmos. Automatización. Aplicaciones a diferentes profesiones. Flujos de Control. Algoritmos. Operadores. Lógica Booleana. Alternativa Condicional.
- **Unidad 5**: Evolución, Análisis y Explotación de Datos. Técnicas para recolección, refinamiento y limpieza de datos. Automatización de análisis de datos con Python. Valores fuera de serie. Predicciones y regresión lineal.

Como se observa, para lograr una curva de aprendizaje suave que partiendo de conceptos básicos de programación llegue hasta herramientas reales de análisis de datos, se reemplazó el lenguaje Scratch empleado en la experiencia piloto por Gobstones[15]. Este lenguaje educativo desarrollado en la Universidad de Quilmes fue concebido para dar los primeros pasos en programación, ofrece una secuencia didáctica escalonada y permite por igual la programación textual y basada en bloques.

4.2 Carga horaria

[15] http://gobstones.github.io/

La materia IPC es cuatrimestral y cuenta con 16 clases de 3 horas cada una, que se dividen en dos partes y abordan los ejes antes mencionados:

	Primera parte	Segunda Parte
Habilidades del pensamiento computacional	16 hs.	
Programación	8 hs.	8 hs.
Análisis de datos		16 hs.

En la **primera parte** de la materia introducimos los conceptos de programación, pensamiento computacional y sus habilidades abordando estos temas desde lo empírico y analizando su presencia en nuestra vida diaria.

En la **segunda parte** aplicamos lo aprendido sobre pensamiento computacional y programación en análisis de datos con un enfoque de aprendizaje basado en proyectos.

Esta distribución horaria varió con respecto a la planteada en el curso piloto y emerge como respuesta antes esta primera experiencia. Aunque originalmente se planteó dedicar más tiempo a la resolución de problemas cotidianos, se decidió reasignarlo a actividades de análisis de datos dado que estas últimas requieren un mayor tiempo de asimilación.

4.3 Enfoque del dictado

4.3.1 Pensamiento computacional

A lo largo de la materia hacemos un recorrido por las habilidades del pensamiento computacional poniendo las mismas en práctica. El objetivo no es dar una definición simplemente sino incorporarlas en el proceso de resolución de problemas. Por cada una de estas:

1. Trabajamos con ejemplos cotidianos que evidencian su presencia más allá de los sistemas informáticos.
2. Vivenciamos la habilidad a través de una actividad lúdica que no involucre el uso de la computadora[16].
3. Analizamos videos, textos, canciones o pinturas desde la óptica de ésta habilidad.
4. Construimos una definición concreta para esta habilidad y discutimos su importancia.
5. Realizamos ejercicios de programación en donde analizamos como su resolución se ve atravesada por dicha habilidad.

[16] Este tipo de actividades son también conocidas como desenchufadas o *unplugged.*

A modo de ejemplo, mencionamos brevemente una actividad que sirve para vivenciar la estrategia de *dividir y conquistar*[17], dentro de la unidad temática de *división en subproblemas*. En ésta entregamos al curso un muñeco y le solicitamos que alguien del público lo esconda. El docente ignorará quien lo posee, pero no así el resto del curso, que debe saber dónde está. Luego preguntamos a cada estudiante si lo tiene hasta llegar al indicado. Es importante que se establezca previamente un orden para preguntar, por ejemplo alfabético o por ubicación espacial dentro del aula.

Luego de encontrarlo se repite el proceso pero cambiamos la estrategia para buscarlo.
Se divide al grupo en dos partes iguales y se le pregunta a un grupo si el integrante que tiene el muñeco pertenece al mismo. En caso afirmativo iteramos el proceso con ese grupo, en caso contrario con el otro. Esta subdivisión se realizará hasta encontrar el muñeco.

El primer tipo de búsqueda es conocido en programación como "búsqueda lineal" mientras que la segunda es conocida como "búsqueda binaria". Estos conceptos nos sirven para trabajar la descomposición en subproblemas mientras discutimos otros conceptos tales como la complejidad algorítmica y los procesos exponenciales.

4.3.2 Programación

El objetivo de Introducción al Pensamiento Computacional es brindar a cada estudiante conocimientos de programación fundamentales y prácticos que les sean útiles en sus distintas disciplinas y aumenten su comprensión sobre los sistemas informáticos. No persigue, por el contrario, la formación de personas que se desempeñen laboralmente desarrollando software.

Por estos motivos el foco no está puesto en el desarrollo de algoritmos complejos sino en el abordaje de conceptos generales aplicados a la resolución de problemas como ser procedimientos, funciones, parámetros, repetición simple, alternativa condicional, expresiones, tipos de datos, lógica booleana, listas y diccionarios.

Para ello trabajamos dentro de la plataforma educativa de programación argentina Mumuki[18] , la cual cuenta con ejercicios autoevaluables creados por el equipo docente de la cátedra. Para dar los primeros pasos e introducir los conceptos de procedimientos (*Figura 1*) y repetición simple (*Figura 2*) programamos con bloques, que nos permite avanzar sin toparnos con potenciales problemas inherentes a la hora de comenzar a programar, como por ejemplo la sintaxis.

[17] Esta estrategia consiste en partir recursivamente los datos de un problema en varios subproblemas similares que operen sobre los subconjuntos, hasta que éstos sean suficientemente simples, y luego combinar los resultados parciales en uno global.
[18] https://mumuki.org

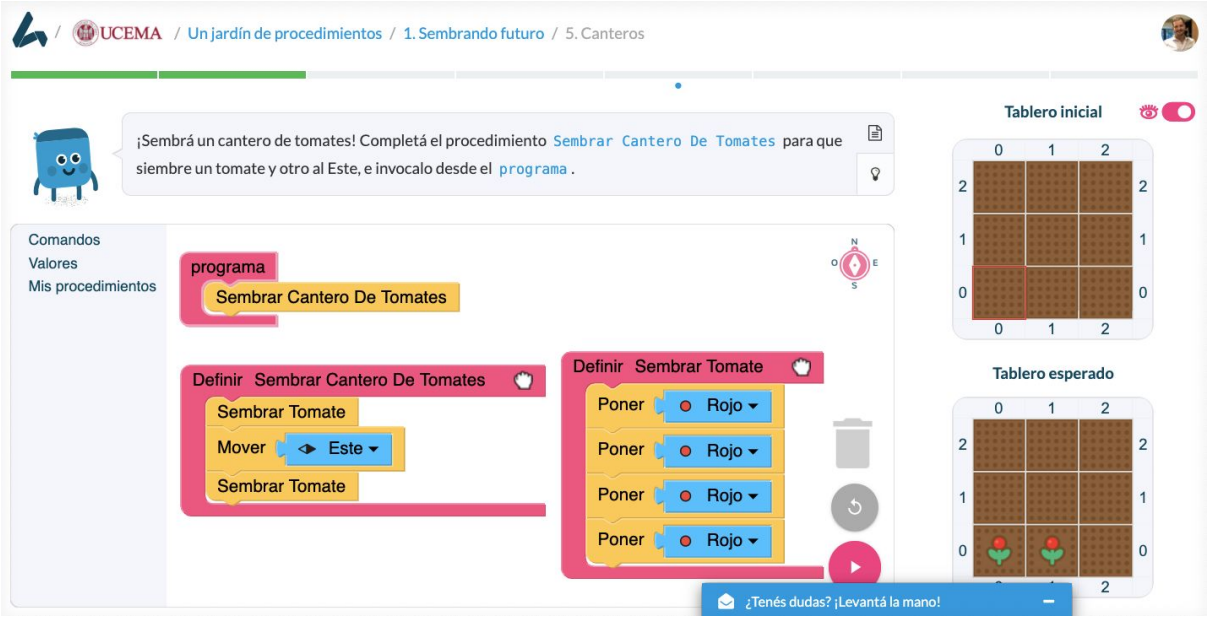


Figura 1: Ejercicio de Gobstones en Mumuki sobre división en subproblemas y procedimientos

Figura 2: Ejercicio de Gobstones en Mumuki sobre identificación de patrones y repetición simple

Luego continuamos utilizando Gobstones, pero realizamos una traducción a programación textual donde seguimos trabajando los conceptos de parámetros, alternativa condicional (*Figura 3*) y funciones.

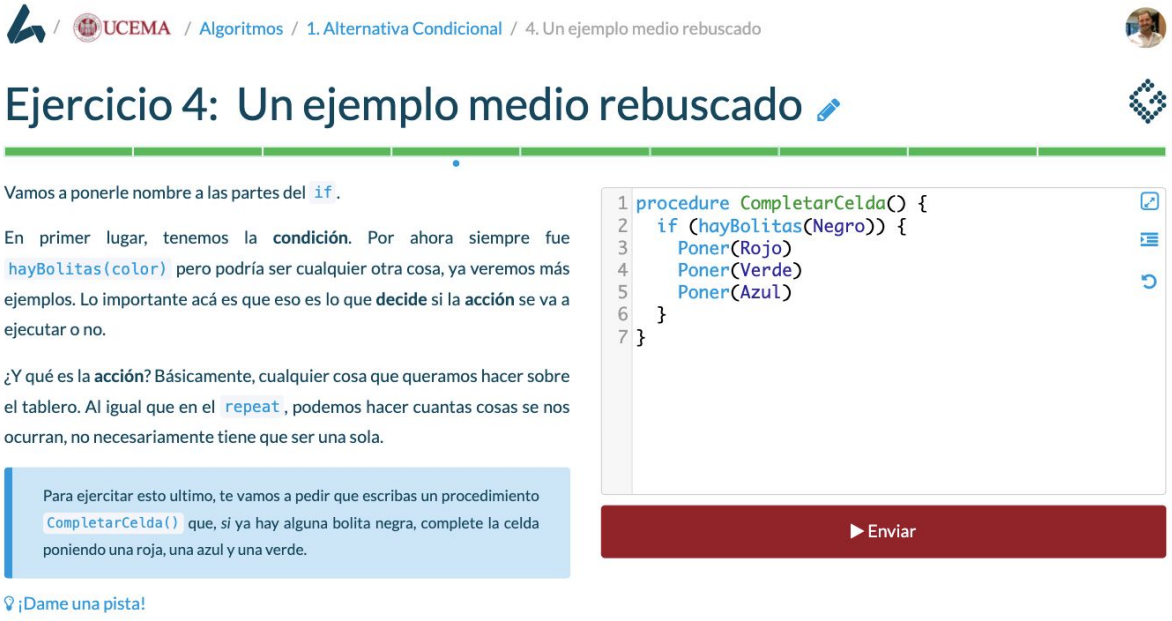


Figura 3: Ejercicio de Gobstones en Mumuki al pasar a la programación textual

4.3.3 Análisis de datos

En este tercer eje la meta es que cada estudiante incorpore los conceptos iniciales de análisis de datos y a partir de distintos lotes pueda elaborar hipótesis para luego verificarlas o refutarlas a partir del uso de gráficos, filtros, agrupaciones, ordenamientos, nociones básicas de estadística y regresiones lineales. Para ello aplicamos lo aprendido sobre pensamiento computacional y programación con un enfoque de aprendizaje basado en proyectos. Además, se incorporan nociones de estadística básica, necesarias para entender diversos aspectos de la manipulación, limpieza y proyección de datos.

Mumuki se sigue utilizando con el objetivo de incorporar nuevos conceptos sobre programación imperativa, al mismo tiempo que abordamos los ya estudiados, desde el lenguaje de programación Python. Sin embargo la principal herramienta utilizada en esta sección es Colab[19], creada por Google y basada en Jupyter[20], que permite desarrollar software y compartirlo fácilmente sin necesidad de preparar un entorno dentro de cada computadora. Dentro de Colab, el alumnado programa utilizando distintas bibliotecas de Python como por ejemplo pandas, numpy y scikit learn (*Figura 4*).

[19] https://colab.research.google.com
[20] https://jupyter.org

```
import pandas as pd
import numpy as np
import sklearn.linear_model as ln
```

Figura 4: Carga de las bibliotecas pandas, numpy y scikit-learn desde Colab

El objetivo es que a partir de la lectura de datos abiertos (*Figura 5*) el curso pueda establecer correlaciones y plantear hipótesis para luego verificarlas o refutarlas aplicando gráficos (*Figura 6*), filtros, ordenamientos, nociones básicas de estadística y regresiones lineales.

```
[ ] recaudacion = pd.read_csv("http://cdn.buenosaires.gob.ar/datosabiertos/datasets/recaudacion-impositiva/recaudacion-impositiva.csv", sep=";")
    recaudacion
```

	periodo	total	impuestos_sobre_ingresos_brutos	alumbrado_barrido_limpieza	impuesto_patentes	impuesto_sellos	plan_facilidades_pago	c
0	ene-97	224.1	135.8	65.2	15.9	NaN	6.8	
1	feb-97	174.1	117.0	17.5	32.1	NaN	6.3	
2	mar-97	180.9	111.8	50.8	11.8	NaN	5.7	
3	abr-97	186.2	126.6	21.4	32.6	NaN	5.2	
4	may-97	223.4	129.5	75.4	12.6	NaN	4.5	
...	...	...	...	...	...	...	...	
217	feb-15	5144.7	3570.1	421.4	677.9	362.7	94.1	
218	mar-15	4707.6	3597.3	475.6	135.2	345.8	102.6	
219	abr-15	5430.9	3957.0	473.4	460.3	407.6	97.7	
220	may-15	5167.1	4046.8	492.4	109.4	397.7	96.1	
221	jun-15	5526.7	3932.2	504.9	490.5	419.3	121.3	

222 rows × 9 columns

Figura 5: Carga de un lote de datos usando pandas

```
recaudacion_nuevo.plot.scatter(x = "ingresos_brutos", y = "abl", figsize=(15,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8bf407a3c8>
```

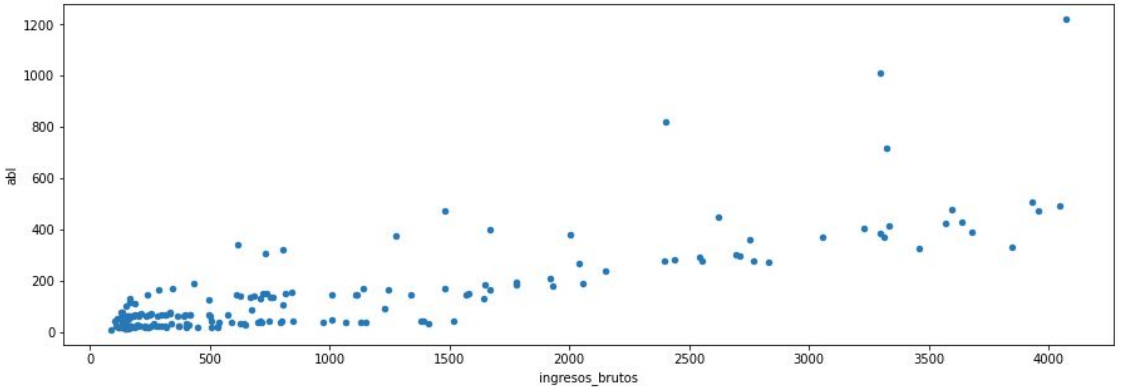


Figura 6: Gráfico de dispersión sobre dos variables de un lote de datos generado con pandas

Con el fin de situar la propuesta educativa, todos los lotes de datos (*datasets*) empleados en los ejemplos y ejercicios de la materia son locales, empleando mayormente datos

nacionales[21] y de Ciudad de Buenos Aires[22] y Provincia de Buenos Aires, donde vive la totalidad del curso. Ocasionalmente se han utilizado también *datasets* de otros países de la región.

4.4 Forma de evaluación

La asignatura cuenta con dos evaluaciones: un trabajo práctico en grupos de 3 a 5 personas de carácter integrador, que se desarrolla iterativamente a lo largo de toda la segunda parte, y un examen individual que se toma en la última clase. Ambas instancias son de carácter obligatorio y su aprobación es necesaria para regularizar la materia.

4.4.1 Trabajo práctico integrador de análisis de datos

Para integrar lo aprendido, el trabajo práctico grupal cuenta con los siguientes objetivos:

- Poner en práctica las herramientas de pensamiento computacional y programación en un caso de análisis de datos.
- Aplicar técnicas y conceptos de análisis de datos para cargar, explorar, limpiar y visualizar datos.
- Utilizar el lenguaje Python, las bibliotecas Pandas y SciKit, y la plataforma Colab para realizar predicciones basadas en datos.
- A partir de la utilización de datos, formular preguntas y darles respuesta, y enunciar hipótesis y confirmarlas o refutarlas.
- Buscar y consultar fuentes acreditadas en Internet para ampliar conocimientos sobre una temática desconocida.

Este trabajo está dividido en tres etapas: exploratoria, de análisis y de profundización.

4.4.1.1 Etapa exploratoria

La primera etapa consiste en explorar el lote de datos: cargarlo, indagar sobre la información que contiene y plantear todas las dudas, preguntas y primeras hipótesis que surjan. Aquí se presentan dos grandes desafíos: por un lado es la primera vez que los grupos consultan un lote de datos (*Figura 9*) buscando interrogantes propios y no respuestas a los planteados por el equipo docente. Por el otro, se enfrentarán a una temática totalmente nueva, siendo menester la indagación sobre la misma para realizar un análisis de datos significativo. Es por ello que se proponen lecturas asociadas a la temática a trabajar y la búsqueda de recursos acreditados en Internet, más allá del correcto uso de las herramientas habituales.

[21] https://datos.gob.ar
[22] https://data.buenosaires.gob.ar

```
casos_covid = ks.read_csv("https://www.datos.gov.co/api/views/gt2j-8ykr/rows.csv?accessType=DOWNLOAD", ",")
casos_covid
```

```
:al/lib/python3.6/dist-packages/IPython/core/interactiveshell.py:2822: DtypeWarning: Columns (10,12,14,16,19,20) have mixed ty
lf.run_code(code, result):
```

ID de caso	Fecha de notificación	Código DIVIPOLA	Ciudad de ubicación	Departamento o Distrito	atención	Edad	Sexo	Tipo	Estado	País de procedencia	FIS
1	2020-03-02T00:00:00.000	11001	Bogotá D.C.	Bogotá D.C.	Recuperado	19	F	Importado	Leve	ITALIA	2020-02-27T00:00:00.000
2	2020-03-06T00:00:00.000	76111	Guadalajara de Buga	Valle del Cauca	Recuperado	34	M	Importado	Leve	ESPAÑA	2020-03-04T00:00:00.000
3	2020-03-07T00:00:00.000	5001	Medellín	Antioquia	Recuperado	50	F	Importado	Leve	ESPAÑA	2020-02-29T00:00:00.000
4	2020-03-09T00:00:00.000	5001	Medellín	Antioquia	Recuperado	55	M	Relacionado	Leve	NaN	2020-03-06T00:00:00.000
5	2020-03-09T00:00:00.000	5001	Medellín	Antioquia	Recuperado	25	M	Relacionado	Leve	NaN	2020-03-08T00:00:00.000
...	...	...	...	...	...	...	...	...	...	...	...
533139	2020-08-14T00:00:00.000	11001	Bogotá D.C.	Bogotá D.C.	Casa	63	F	En estudio	Leve	NaN	2020-08-14T00:00:00.000
533140	2020-08-16T00:00:00.000	11001	Bogotá D.C.	Bogotá D.C.	Casa	41	F	En estudio	Leve	NaN	2020-08-16T00:00:00.000
533141	2020-08-16T00:00:00.000	11001	Bogotá D.C.	Bogotá D.C.	Hospital	77	M	En estudio	Moderado	NaN	2020-08-11T00:00:00.000
533142	2020-08-17T00:00:00.000	11001	Bogotá D.C.	Bogotá D.C.	Casa	62	F	En estudio	Leve	NaN	2020-08-11T00:00:00.000
533143	2020-08-16T00:00:00.000	11001	Bogotá D.C.	Bogotá D.C.	Hospital	89	M	En estudio	Moderado	NaN	2020-08-08T00:00:00.000

Figura 9: Extracto de la carga de un lote de datos para el trabajo práctico integrador

4.4.1.2 Etapa de análisis

En la segunda etapa, tras hacer las primeras lecturas del lote de datos y las fuentes de respaldo, y anotar todas las preguntas, dudas e hipótesis que surgen, procedemos a discutirlas y responderlas. Cuando es posible, se invita al curso a especialistas para ampliar los conceptos de la disciplina. Como consecuencia de este proceso se evidencian nuevas preguntas e hipótesis, que sirven como direcciones de análisis que cada grupo debe recorrer.

En esta etapa los grupos comienzan a realizar las limpiezas (*Figura 10*), transformaciones, visualizaciones y las consultas a los datos que permiten realizar sus primeras predicciones y corroborar o refutar sus hipótesis iniciales al mismo tiempo que plantean nuevas.

▾ 4.1. Eliminando NaNs

El primer problema con el que nos encontraremos es que algunos de los valores de *poblacion* son nulos (NaN). Si quisieramos eliminar todas las filas que tengan en esta columna valores nulos, podríamos hacer lo siguiente:

```
# notna y notnull son sinónimos, hacen lo mismo, podés usar cualquiera de los dos
enfermedades_validas = enfermedades[enfermedades.Poblacion.notna()] #esto se puede hacer para cada uno de los virus
enfermedades_validas
```

	Fecha	Numero	Sexo	Poblacion	AdenoV	RSV	INFA	INFB	PIV1	PIV2	PIV3	Edad(meses)
0	03/01/11	1	F	CABA	-	-	-	-	-	-	-	96.0
1	03/01/11	2	F	LA FERRERE	-	-	-	-	-	-	-	4.0
2	03/01/11	3	M	VILLA CELINA	-	-	-	-	-	-	-	8.0
3	03/01/11	4	M	VILLA CELINA	-	-	-	-	-	-	-	8.0
5	03/01/11	6	M	CABA	-	-	-	-	-	-	-	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...
6722	03/07/12	6723	M	CIUDAD EVITA	-	-	-	-	-	-	-	2.0
6723	03/07/12	6724	F	CABA	-	-	-	-	-	-	-	60.0
6724	03/07/12	6725	M	CABA	-	-	-	-	-	-	-	48.0
6725	03/07/12	6726	M	TRES DE FEBRERO	-	-	-	-	-	-	-	4.0
6726	03/07/12	6727	M	CABA	NM	NM	NM	NM	NM	NM	NM	31.2

4650 rows × 12 columns

Figura 10: Filtrado de un lote de datos

4.4.1.3 Etapa de profundización

En la etapa final la idea es refinar el análisis realizado en la segunda etapa, buscando qué se puede mejorar del mismo para llegar a un resultado superador.

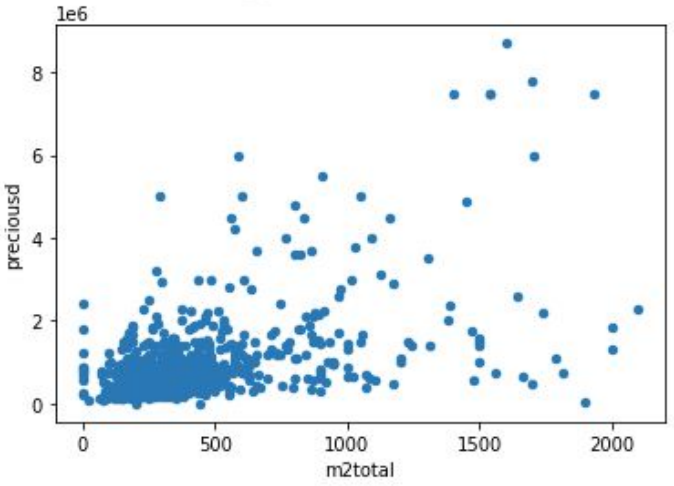
4.4.2 Parcial

El parcial es una instancia de evaluación individual por computadora en la cual presentamos un lote de datos sobre una temática no trabajada previamente y realizamos algunas preguntas cuantitativas y cualitativas que deberán ser respondidas mediante el uso de la herramientas de programación vistas, empleando la plataforma Colab.

A la hora de corregir el mismo hacemos hincapié en la presencia de las habilidades de pensamiento computacional, las buenas prácticas de programación y la correcta aplicación de los conceptos de análisis de datos vistos en la materia. Son por tanto criterios de evaluación la descomposición en subproblemas, la elaboración de algoritmos, la creación de procedimientos, la realización de gráficos que respalden la información obtenida (*Figura 11*), la utilización de medidas estadísticas para limpiar los datos, las predicciones que pueden realizar y la fiabilidad sobre las mismas tanto cuantitativa como cualitativamente.

```
[12] terrenos_validos.plot.scatter(x = "m2total", y = "preciousd")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fba1ab7d5f8>
```




```
[13] def crear_eje_x(dataset, nombre_columna):
       return np.array(dataset[nombre_columna]).reshape(-1, 1)

     def crear_eje_y(dataset, nombre_columna):
       return np.array(dataset[nombre_columna])

     def predecir(modelo, x):
       x_para_predecir = np.array([x]).reshape(-1, 1)
       return modelo.predict(x_para_predecir)

     def calcular_correlacion(modelo, dataset, columna1, columna2):
       eje_x = crear_eje_x(dataset, columna1)
       eje_y = crear_eje_y(dataset, columna2)
       modelo.fit(eje_x, eje_y)
       return modelo.score(eje_x, eje_y)
```

Figura 11: Uso de procedimientos para simplificar el proceso de regresión lineal

5 Trabajo futuro

Con el objetivo de posibilitar la reproducibilidad de estas actividades en otros contextos, es de sumo interés para el equipo docente de la materia sistematizar y documentar las didácticas empleadas. Queda como trabajo futuro un desarrollo más extenso y completo tanto de las actividades desenchufadas como de los materiales audiovisuales analizados.
Además, el equipo está actualmente desarrollando y realizando las primeras pruebas de herramientas educativas para simplificar el análisis de datos, que no han sido cubiertas en esta memoria pero serán oportunamente publicadas.

6 Conclusiones

Aunque inicialmente la materia encontró sus bases en experiencias previas realizadas en otras universidades, los aprendizajes del curso piloto y los nueve dictados subsiguientes moldearon su carácter. Así, innovaciones como la creación de nuevas didácticas, el uso de tecnologías argentinas y lotes de datos abiertos locales, e incluso la construcción de herramientas nacidas en el seno de la materia, se convirtieron en sus rasgos distintivos. Sin embargo, lejos de ser una materia acabada, su evolución es constante.

El contexto atípico del año 2020, hizo necesaria la revisión de su modalidad. Esto lejos de ser un impedimento para su dictado, se convirtió en una nueva oportunidad de mejora que posibilitó la creación de nuevas actividades así como la adecuación de las existentes. Al mismo tiempo, la materia sirvió para canalizar nuevas discusiones referentes acerca del rol de la tecnología en general y la minería de datos en particular y el ejercicio de la ciudadanía digital.

Si bien hemos encontrado un desafío en lograr una materia actualizada y enriquecedora para perfiles heterogéneos, podemos afirmar que la materia ha resultado de gran interés y utilidad para sus estudiantes. Ésto lo observamos no sólo dentro del aula sino también fuera de ella. En varias ocasiones, por ejemplo, sobre el final de la cursada nos han solicitado material adicional sobre programación y análisis de datos y nos han compartido experiencias sobre la puesta en práctica de los conceptos aprendidos en proyectos personales.