
Formulario de Aprobación Curso de Posgrado

Asignatura:

Introducción a la Filosofía de la Ciencia de la Computación.

Profesor de la asignatura ¹:

Alejandro Chmiel Prof. Asistente (Gr. 2), Instituto de Filosofía de la FHCE
Guillermo Nigro (profesor de filosofía (ANEP))

Profesor Responsable Local ¹:

Dra. Sylvia da Rosa, Prof. Agregada (Gr. 4), Instituto de Computación, FING

Horas Presenciales: 30

Horas de dedicación del estudiante: 90

Nº de Créditos: 6

(de acuerdo a la definición de la Udelar, un crédito equivale a 15 horas de dedicación del estudiante según se detalla en el ítem metodología de la enseñanza)

Público objetivo y Cupos:

Cupo máximo 20 personas.

El curso se dicta en el marco de las actividades del Núcleo Interdisciplinario *Filosofía de la Ciencia de la Computación*, y se ofrecerá a:

Estudiantes de posgrado o grado de Ingeniería en Computación.

Estudiantes de posgrado o grado de Filosofía.

Estudiantes y profesores del profesorado de informática.

A estudiantes y profesores vinculados al Espacio Interdisciplinario de la UDELAR.

Objetivos:

Introducir al estudiante en los problemas de la filosofía de la ciencia de la computación, desde su dimensión ontológica, epistemológica, y metodológica. Los problemas, en la medida en que sea posible, se abordarán desde una perspectiva interdisciplinaria.

Conocimientos previos exigidos: Ninguno.

Conocimientos previos recomendados: Un curso básico de lógica. Conocimientos básicos de programación. Lectura de inglés.

Forma de evaluación:

Los participantes deberán responder una serie de preguntas y ejercicios, por cada unidad y elaborar una presentación oral hacia el final del curso, sobre tema a determinar.

Temario:

Unidad 1. Introducción a los problemas.

1. ¿Qué es la computación? Perspectivas generales.
2. La noción de programa.
3. Presentación de los tres principales paradigmas desde los cuales se interpreta la disciplina: racionalista, científico y tecnológico.

Unidad 2. Introducción a los problemas ontológicos.

1. ¿Qué es un programa? Nivel físico y nivel simbólico.
2. Compromiso ontológico.
3. Breve desarrollo histórico de los lenguajes de programación.
4. Distinción entre distintos tipos de lenguajes de programación.
5. Lenguajes de programación y sus compromisos ontológicos.
6. Relación entre el programa y la máquina (implementación física).
7. Formalismo y platonismo.

Unidad 3. Introducción a los problemas epistemológicos.

1. Introducción al problema del conocimiento.
2. ¿Qué podemos conocer acerca de un programa?
3. ¿Qué significa que un programa es correcto?
4. Problemas de complejidad vinculados a la evaluación de la corrección.

Unidad 4. Tópicos especiales.

1. Problemas filosóficos en torno al concepto de *máquina*.
 2. Problemas filosóficos de la inteligencia artificial.
-

Bibliografía:

Amnon H Eden. "Three Paradigms of Computer Science." *Minds and Machines*, Special issue on the Philosophy of Computer Science, Vol. 17, No. 2 (Jul. 2007), pp. 135-167. London: Springer.

Amnon H. Eden, Raymond Turner. "Problems in the ontology of computer programs." *Applied Ontology* Vol. 2, No. 1 (2007), pp. 13-36. Amsterdam: IOS Press.

DeMillo, R.A., Lipton, R.J. and Perlis, A.J., 1979, "Social Processes and Proofs of Theorems and Programs", *Communications of the ACM* 22(5): 271-280.

Fetzer, J.H., 1988, "Program Verification: The Very Idea", *Communications of the ACM* 31(9): 1048-1063.

Chalmers, D. Does a Rock Implement Every FiniteState Automaton? (1996)

Colburn, T., 2004, "Methodology of Computer Science", *The Blackwell Guide to the Philosophy of Computing and Information*, Luciano Floridi (ed.), Malden: Blackwell, pp. 318-326.

Colburn, T., and Shute, G., 2007, "Abstraction in Computer Science", *Minds and Machines* 17(2): 169-184.

Copeland, B. Jack, "The Modern History of Computing", *The Stanford Encyclopedia of Philosophy* (Fall 2008 Edition), Edward N. Zalta (ed.), URL = <http://plato.stanford.edu/archives/fall2008/entries/computing-history/>.

Copeland, B. Jack, 2008, "The Church-Turing Thesis", *The Stanford Encyclopedia of Philosophy* (Fall 2008 Edition), Edward N. Zalta (ed.), URL = <http://plato.stanford.edu/archives/fall2008/entries/church-turing/>.

Copeland, B. Jack, 2004, "Computation", *The Blackwell Guide to the Philosophy of Computing and Information*, Luciano Floridi (ed.), Malden: Blackwell, pp. 3-17.

Fetzer, J.H., 1988, "Program Verification: The Very Idea", *Communications of the ACM* 31(9): 1048-1063.

Floridi, Luciano, 2004. "Information", *The Blackwell Guide to the Philosophy of Computing and Information*, Luciano Floridi (ed.), Malden: Blackwell, pp. 40-62.

Floridi, Luciano 2007, "Semantic Conceptions of Information", *The Stanford Encyclopedia of Philosophy* (Spring 2007 Edition), Edward N. Zalta (ed.), URL = <http://plato.stanford.edu/archives/spr2007/entries/information-semantic/>.

Gandy, R. (1980) Church's Thesis and principles for mechanisms; in: The Kleene Symposium (edited by J. Barwise, H.J. Keisler and K. Kunen), NorthHolland, 123-148.

Hoare, C.A.R., 1969, "An axiomatic basis for computer programming". *Communications of the ACM* 12(10):576-585.

Moor, J.H., 1978, "Three Myths of Computer Science", *The British Journal for the Philosophy of Science* 29(3): 213-222.

Rapaport, W.J., 2005b, "Implementation is Semantic Interpretation: Further Thoughts." *Journal of Experimental and Theoretical Artificial Intelligence* 17(4): 385-417.

Sieg, W. (1994). *Mechanical Procedures and Mathematical Experience. Mathematics and Mind*. A. George. New York, Oxford University Press: 71-117.

Smith, B.C., 1996, "Limits of Correctness in Computers", *Computerization and Controversy*, Kling, R. (ed.), Morgan Kaufman, pp. 810-825.

Turner, Raymond, 2007, "Understanding Programming Languages". *Minds and Machines* 17(2): 129-133.

Turner Raymond and Eden, Amnon. "Philosophy of computer science." In *The Stanford Encyclopedia of Philosophy* (Winter 2008 edition), Edward N. Zalta (ed.).

[Richard L. Wexelblat](#) (ed.), 1981, *History of Programming Languages*, Academic Press.