Pensamiento Computacional

Organización del curso

Identificación

Nombre del curso: Pensamiento Computacional
Componente: Computacional
Programa académico: Ingeniería de Sistemas

- Nivel académico: Pregrado
- Número de créditos: 3
- Tipo de curso: Teórico práctico
- Número de semanas: 16
- Horas por semana: 5

Propósitos de formación

1. Desarrollar el pensamiento computacional para la solución de problemas reales
2. Apropiarse de las TIC como herramientas en la solución de problemas prácticos
3. Desarrollar habilidades de abstracción para la construcción de modelos y resolución de problemas
4. Desarrollar soluciones lógicas y algorítmicas a problemas computacionales
5. Identificar escenarios de aplicación del pensamiento computacional para diferentes áreas de formación

Justificación

El avance tecnológico y su incidencia en escenarios cada vez más cotidianos obligan a entender y usar los recursos computacionales en la resolución de problemas, que vistos desde afuera, y sin conocimientos informáticos, la aparente complejidad de la programación de computadores y el hecho de soportarse en las matemáticas, suele ser de entrada una indisposición para éste fin, ya sea como amateur o profesional.

El pensamiento computacional en sí misma es una competencia para la vida y para el desarrollo de cualquier campo de formación, no solo para programadores en contextos informáticos, sino también en la solución de problemas de la vida diaria, en la forma de pensar e interpretar el mundo, hasta convertirse en una línea transversal en todos los niveles y áreas de formación.

El proceso de aprendizaje en programación se debe dar desde lo abstracto a lo concreto, quizá el principal problema de la programación sea que se ha quedado en lo abstracto sin avanzar hacia lo concreto, sin resultados visibles sobre situaciones reales en tiempo real, para lo que se debe romper el aprendizaje plano tradicional de código y sustituirse por aprendizajes a partir de experiencias con

el uso de herramientas de programación gráfico y escenarios de prácticas físicas, escenarios en los que prima el trabajo participativo y colaborativo basado en proyectos.

Con el desarrollo del pensamiento computacional en los estudiantes del programa de ingeniería de sistemas de la Universidad de Pamplona se pretende formar a los innovadores tecnológicos del futuro.

Competencias

Generales

1. Promover el trabajo colaborativo para desarrollar actividades y el aprendizaje basado en proyectos.
2. Enfocar la atención del alumno en problemas abiertos y soluciones abiertas, así como desarrollar un pensamiento crítico, habilidades de liderazgo, comunicación, responsabilidad social, creatividad e innovación.

Específicas

1. Entender qué aspectos de un problema son susceptibles a resolverse con técnicas de la computación.
2. Implementar modelos y simulaciones para generar conocimiento y probar hipótesis en cualquier área del conocimiento.
3. Reconocer oportunidades en la industria para aplicar las técnicas de la computación.
4. Explorar nuevas formas de expresión creativa por medio de las técnicas de la computación.
5. Extraer y descubrir información almacenada en grandes conjuntos de datos.
6. Explicar los problemas y soluciones en términos computacionales.

Metodología

Apoyados en los recursos tecnológicos el aprendizaje se logra de forma práctica, desarrollando proyectos de casos reales, en tareas que suelen considerarse complejas, en retos de formación que despierten la pasión por los resultados obtenidos, y basados en proyectos que pueden desarrollar la creatividad en búsqueda de nuevas experiencias, aumentando los niveles de complejidad “sin darse cuenta”.

Los estudiantes empiezan a aprender a programación, si saber que están programando, para llevarlos luego a la formalización del código, la metodología se basa en proyectos y cada proyecto incentiva a nuevos retos y al uso de nuevas herramientas que vayan aumentando los niveles de dificultad y complejidad, en el que se prioriza la exploración y aprendizaje autónomo, logrando que el estudiante aprenda de acuerdo a sus intereses y creatividad.

Todo proyecto parte de una corta fundamentación teórica, instrucciones y recomendaciones, definición de materiales, modelados, montajes, programación, y terminan con identificación de nuevos proyectos.

El proceso se lleva a cabo con asesoría y acompañamiento del docente, con el uso de recursos tecnológicos lógicos y físicos, y apoyados con ambientes de aprendizaje como laboratorios prácticos y entornos virtuales como Moodle.

Actividades de evaluación

1. Participación, logros y retos dentro de las actividades prácticas desarrolladas.
2. Definición y formulación de proyectos libres.
3. Avances de proyecto: uno para cada corte.
4. Proyecto final: Entrega y socialización de proyectos.
5. Pruebas escritas: fundamentación, algoritmos y código.

Calendario

- Corte 1: Semana 6
  15% trabajo en clase y avance1 del proyecto
  20% prueba escrita
- Corte 2: Semana 11
  15% trabajo en clase y avance2 del proyecto
  20% prueba escrita
- Corte 3: Semana 16
  10% trabajo en clase y socialización final del proyecto
  20% prueba escrita

Contenidos

Unidad 1: Fundamentación

1. Pensamiento computacional (abstracción, modelado, simulación, etc…)
2. Aprendizaje basado en proyectos
3. Programar “sin código”: programación como lenguaje de creación y expresión
4. Noción de algoritmos
5. Noción de programación secuencial
6. Noción de programación estructurada
7. Noción de programación modular (funciones)
8. Noción de programación orientada a objetos
9. Noción de paralelismo
10. Introducción a Stratch (los contenidos se van desarrollando en los proyectos)
11. Introducción a Arduino (los contenidos se van desarrollando en los proyectos)
12. PROYETOS (Lego, Scratch, Arduino)

Unidad 2: Programación

1. Definición y estructura de un Algoritmo
2. Representación de los algoritmos (Diagramas de flujo, seudocódigo y código)
3. Tipos de datos, variables, constantes, contadores y acumuladores
4. Lenguajes de programación
5. PROYECTOS (Lego, Scratch, Arduino, C++)

Unidad 3 Estructuras secuenciales

1. Instrucciones
2. Entradas y salidas
3. Expresiones
4. Operaciones
5. PROYECTOS (Lego, Scratch, Arduino, C++)

Unidad 4: Estructuras de selección, condicionales o de control

1. Sentencia if
2. Sentencia if-then-else
3. Setencia switch
4. Estructuras de selección anidadas
5. PROYECTOS (Lego, Scratch, Arduino, C++)

Unidad 5: Estructuras de iteración o repetitivas

1. Bucle for
2. Bucle while
3. Estructuras de iteración anidadas
4. PROYECTOS (Lego, Scratch, Arduino, C++)

Unidad 6: Estructuras modulares

1. Funciones
2. Parámetros
3. Llamado para la ejecución de funciones
4. Llamado entre funciones
5. PROYECTOS (Lego, Scratch, Arduino, C++)

Contenidos secundarios – En el desarrollo de los proyectos

Scratch:

1. Desarrollo del pensamiento computacional con Scratch
2. Abstracción (cómo comunicar ideas complejas de forma sencilla y descomponer problemas con lógica)
3. Integración de contenidos multimedia (texto, imágenes, sonido, datos, gráficos)
4. Desarrollo de objetos y bloques funcionales (objetos y programas)
5. Programas interactivos (eventos y gestión de eventos)
6. Conceptos fundamentales de programación (decisiones, bucles, variables, funciones, ejecución secuencial y paralelismo)

Computación Física:

1. Fundamentos de electrónica
2. Programación digital
3. Programación análoga
4. Programación serial

Recursos

Herramientas para el desarrollo

- Scratch
- IDE Arduino
- Arduino y Scratch: S4A (otras: mBlock, Bitbloq, ScratchX, Ardublock, miniBloq)
- AppInventor
- Dev-C++ (otras: Code::Blocks, NetBeans, Eclipse, etc…)

Herramientas de diseño

- Crocodile clip
- Fritzing
- Día - UML

Recursos físicos

- PC
- Tarjeta de desarrollo Arduino UNO
- Componentes electrónicos (leds, sensores, motores, etc.)

Recursos Didácticos

- Guías
- Tutoriales
- Videotutoriales
- Materiales web

Ambientes de aprendizaje

- Laboratorios (talleres prácticos)
- Ambientes virtuales de aprendizaje: Moodle

Bibliografía

- El Pensamiento Computacional en la Enseñanza Obligatoria (Computhink), Febrero 2017. Instituto Nacional de Tecnologías Educativas y de Formación del Profesorado (INTEF) – Departamento de Proyectos Europeos
  http://blog.educalab.es/intef/wp-content/uploads/sites/4/2017/02/2017_0206_CompuThink_JRC_UE-INTEF.pdf
- Pensamiento Computacional un Aporte para la Educación de Hoy
- Gurises Unidos en el marco del proyecto Robotic-Pensamiento Computacional de Fundación Telefónica – Movistar. 2017
  http://www.gurisesunidos.org.uy/wp-content/uploads/2017/11/PensamientoComputacional.pdf

- Pensamiento Computacional (PC) en educación escolar. Caja de herramientas para líderes, primera edición. Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE) 2011 http://eduteka.icesi.edu.co/pdfdir/PensamientoComputacional1.pdf
- Porcuna López, Pedro. Robótica y domótica básica con Arduino. Ediciones de la U
- Monk, Simo. Programación de Arduino. Introducción a Sketches. Editorial: Anaya
- Trejos Buriticá, Omar Iván. Lógica de Programación. Ediciones de la U Limitada.
- Joyanes. Fundamentos Generales de Programación Mcgraw Hill. 2012
- Fernández Huerta, Iván. Fundamentos Básicos de Programación: Aplicación Práctica con scratch y phyton. Delta Publicaciones
- Pérez-Paredes, Pascual y Zapata Ros, Miguel. El pensamiento computacional, análisis de una competencia clave (Spanish Edition) (Spanish) Paperback – May 10, 2018
- Cárdenas González, y otros. Pensamiento lógico computacional (Spanish Edition) Kindle Edition. Editorial Digital del Tecnológico de Monterrey (July 3, 2015)
- Ojeda Rojas, José. Iniciate en Programacion con Scratch: creando un juego con Scratch May 17, 2017
- Anta, Andrés. ¡Hazlo fácil! ...y dominarás la programación en Scratch: Programando juegos 2ª entrega (Programación lúdica y educativa en Scratch) February 15, 2017

En la Biblioteca Universidad de Pamplona:

- Torrente Artero Óscar, Arduino: Aplicaciones en Robótica, Mecatrónica e Ingenierías, Edición 1, Alfaomega, México, 2013
- Cid Manjaraz Jaime, Reyes Cortés Fernando. Arduino: Aplicaciones en Robótica, Mecatrónica e Ingenierías. Alfaomega México 2015
- Jiménez Murillo José Alfredo, Alvarado Zamora Laura Nelly, Jiménez Hernández Eréndira. Fundamentos de programación: diagramas de flujo, diagramas n-s, pseudocódigo y java. Alfaomega México 2016