PROGRAMA DE ASIGNATURA	
ASIGNATURA: Algoritmos y Programación	**AÑO:** 2021
CARACTER: Obligatoria	**UBICACIÓN EN LA CARRERA:** 1° año 1° cuatrimestre
CARRERA: Licenciatura en Matemática Aplicada	
REGIMEN: Cuatrimestral	**CARGA HORARIA:** 120 Horas.

FUNDAMENTACIÓN Y OBJETIVOS

De los diversos modelos de lenguajes de programación existentes en la actualidad, el de la programación imperativa es el más antiguo y aún hoy el más ampliamente utilizado. Según este paradigma y en un sentido amplio del término, un programa se compone de instrucciones para ser ejecutadas por una máquina. La ejecución de una instrucción lleva la máquina de un estado a otro. Las instrucciones que pueden emplearse en la construcción de un programa dependen de la máquina que desea programarse, deben ser instrucciones que ella sea capaz de ejecutar. Esto da lugar a una amplia variedad de lenguajes de programación imperativos, desde lenguajes de propósito general, capaces de programar todo tipo de computadoras a otros destinados a máquinas específicas, físicas o virtuales.

A pesar de la diversidad de lenguajes de programación imperativos de propósito general, existen ciertas características comunes a todos ellos: la posibilidad de acceder y modificar porciones de memoria de una manera amigable (variable y asignación), de representar datos estructurados (definición de tipos y/o clases), de tomar decisiones (condicionales), de repetir instrucciones (ciclos), de programar en forma estructurada (descomponiendo en bloques, funciones, procedimientos, módulos, etc.), de interactuar con un usuario (entrada/salida), entre otras.

El propósito de este curso es ofrecer un abordaje gradual a estas características generales a través de la presentación de los conceptos y la ejercitación continua, introduciendo simultáneamente técnicas de diseño, buenas prácticas de programación, identificando vicios comunes y proponiendo métodos para razonar sobre los programas.

Entre los objetivos se busca asimilar conceptos fundamentales de programación imperativa mediante la ejercitación, conocer técnicas habituales de programación, incorporar buenas prácticas, reconocer la necesidad de comprobar el buen funcionamiento de los programas y familiarizarse con técnicas para ello, adquirir habilidad en el abordaje computacional de problemas numéricos simples.

CONTENIDO

Programación con bloques.

Programación elemental en code.org: Movimientos en un tablero. Instrucciones elementales. Composición secuencial. Ciclo repetir n veces. Ciclo repetir hasta. El problema de la no terminación. Condicional simple. Condicional. Primeros valores: números y colores. Gráficos turtle (code.org). Anidamiento de ciclos de profundidad 2 y 3. Ciclo mientras.
Programación estructurada en code.org: Funciones predefinidas. Parámetros. Ciclo para. Definición de funciones. Funciones y más funciones. Variables y su alcance.

Transición a lenguaje de programación de propósito general.

Python: Módulo turtle. Representación de los programas hechos en code.org. Instrucciones elementales. Composición secuencial. Ciclo for. Ciclo while. El problema de la no terminación. Condicional. Forma general del condicional. Valores numéricos. Colores. Valores aleatorios. Ciclos. Definición de funciones. Normas de estilo.

Construcciones básicas.

Variables, constantes, literales, expresiones. Asignación, asignación múltiple. Iteración. Ciclo for, índice, rango, cuerpo. Funciones: declaración o definición, invocación o llamada. Parámetros y argumentos. Valores numéricos enteros y reales, valores lógicos. Errores de representación de los valores reales. Operadores aritméticos, operadores relacionales y operadores lógicos. Expresiones aritméticas, definición de funciones aritméticas simples. Expresiones lógicas, definición de funciones lógicas simples. Instrucciones condicionales, forma general. Buenas y malas prácticas de programación con condicionales. Ejemplos: año bisiesto, fecha válida. Comportamiento no conmutativo de la conjunción y la disyunción. Determinación el día de la semana de una fecha dada.

Secuencias.
Programando sobre secuencias (listas o arreglos) y cadenas de caracteres. Operaciones básicas. Recorrida de secuencias, conteo de ocurrencias de elementos en secuencias. Mínimo y máximo de secuencias. Posición del mínimo. Posición del mínimo a partir de un índice dado. Determinar si una secuencia está ordenada. Ordenación por selección.

Iteración.
Ciclo while, condición, cuerpo. Ejemplos: algoritmo de división, chequeo de primalidad, factores primos, factorización, raíz entera, mcd, otros ciclos. Buenas y malas prácticas de programación con ciclos. Corrección parcial de un programa: tipos, precondición, estado, invariante, demostración del invariante, poscondición. Ejemplos: algoritmo de división, chequeo de primalidad, factores primos, raíz entera, mcd, factorial. Variantes y corrección total. Especificación e implementación.

Tipos abstractos de datos.
Definiciones recursivas. Recursión sobre números naturales. Ejemplos: factorial, fibonacci, potenciación, división entera, mcd, mcd extendido, número combinatorio. Recursión e inducción. Corrección. Iteración vs recursión. Recursión sobre secuencias. Ejemplos: mínimo, posición del mínimo, búsqueda lineal, búsqueda binaria. Técnicas de diseño top-down y bottom-up.

Entrada/salida.
Interfaz con el usuario. Cadenas de caracteres y de otros tipos de datos. Lectura y escritura de archivos.

Programación funcional.
Un vistazo de programación funcional elemental. Recursión sobre listas. Tipos abstractos de datos en programación funcional.

BIBLIOGRAFÍA

BIBLIOGRAFÍA BÁSICA

Introduction to programming using Python. Daniel Liang. Armstrong Atlantic State University, 2013.

Tutorial oficial de Python: Tutorial de Python — documentación de Python - 3.9.1

Curso acelerado de code.org https://studio.code.org/s/20-hour

Thinking Functionally with Haskell. Richard Bird. Cambridge University Press, 2015.

BIBLIOGRAFÍA COMPLEMENTARIA

Aprenda a pensar como un programador con Python. Allen Downey, Jeffrey Elkner, Chris Meyers. Green Tea Press, 2012.

Introduction to computation and programming using Python. John V. Guttag. The MIT Press. Revised and Expanded Edition, 2012.

Algoritmos de Programación con Python. R. Wachenchauzer, M. Manterola, M. Curia, M. Medrano, N. Paez. uniwebsidad.com

Python Tutorial. w3schools.com.

Blockly: A JavaScript library for building visual programming editors. https://developers.google.com/blockly

Programming Fundamentals: A Modular Structured Approach, 2nd Edition, Kenneth Leroy Busbee and, Dave Braunschweig (creative commons.)

Structured Programming, O.-J. Dahl, E. W. Dijkstra, C. A. R. Hoare, Academic Press, London,1972.

Python Practice Book, 2019, Anand Chitipothu (creative commons).

https://docs.python.org/

Think Python, 2012 o 2016, Allen B. Downey.

Introduction to computation and programming using Python. John V. Guttag. Revised and Expanded Edition, 2012

EVALUACIÓN

FORMAS DE EVALUACIÓN

Deberán entregarse diez (10) actividades de programación, con periodicidad semanal. Se hará una devolución individual en los días subsiguientes a las entregas.
-Examen final

REGULARIDAD

Aprobar al menos seis (6) de las diez (10) actividades de programación semanales.

PROMOCIÓN

Aprobar todas las evaluaciones parciales con una nota no menor a seis (6), y obteniendo un promedio no menor a siete (7). Y aprobar un coloquio.