Sílabo

Malla 2021

DEPARTAMENTO

Departamento de Computer Science

CURSO

Programación I

MALLA

2021

MODALIDAD

Blended

CREDITOS

4

REGLAS INTEGRIDAD ACADÉMICA

Todo estudiante matriculado en una asignatura de la Universidad de Ingeniería y Tecnología tiene la obligación de conocer y cumplir las reglas de integridad académica, cuya lista a continuación es de carácter enunciativo y no limitativo, ya que el/la docente podrá dar mayores indicaciones:

1. La copia y el plagio son dos infracciones de magnitud muy grave en la Universidad de Ingeniería y Tecnología (UTEC) conforme a lo establecido en el Reglamento de Disciplina de los Estudiantes. Tienen una sanción desde 2 semestres de suspensión hasta la expulsión.
2. Si se identifica la copia o plagio en evaluaciones individuales, el/la docente puede proceder a anular la evaluación.
3. Si la evaluación es personal o grupal-individual, la interacción entre equipos o compañeros se considera copia o plagio, según corresponda. Si la evaluación calificada no indica que es grupal, se presume que es individual.
4. La copia, plagio, el engaño y cualquier forma de colaboración no autorizada no serán tolerados y serán tratados de acuerdo con las políticas y reglamentos de la UTEC, implicando consecuencias académicas y sanciones disciplinarias.
5. Aunque se alienta a los estudiantes a discutir las tareas y trabajar juntos para desarrollar una comprensión más profunda de los temas presentados en este curso, no se permite la presentación del trabajo o las ideas de otros como propios. No se permite el plagio de archivos informáticos, códigos, documentos o dibujos.
6. Si el trabajo de dos o más estudiantes es sospechosamente similar, se puede aplicar una sanción académica a todos los estudiantes, sin importar si es el estudiante que proveyó la información o es quien recibió la ayuda indebida. En ese sentido, se recomienda no proveer el desarrollo de sus evaluaciones a otros compañeros ni por motivos de orientación, dado que ello será considerado participación en copia.
7. El uso de teléfonos celulares, aplicaciones que permitan la comunicación o cualquier otro tipo de medios de interacción entre estudiantes está prohibido durante las evaluaciones o exámenes, salvo que el/la docente indique lo contrario de manera expresa. Es irrelevante la razón del uso del dispositivo.
8. En caso exista algún problema de internet durante la evaluación, comunicarse con el/la docente utilizando el protocolo establecido. No comunicarse con los compañeros dado que eso generará una presunción de copia.
9. Se prohíbe tomar prestadas calculadoras o cualquier tipo de material de otro estudiante durante una evaluación, salvo que el/la docente indique lo contrario.
10. Si el/la docente encuentra indicios de obtención indebida de información, lo que también implica no cumplir con las reglas de la evaluación, tiene la potestad de anular la prueba, advertir al estudiante y citarlo con su Director de Carrera. Si el estudiante no asiste a la citación, podrá ser reportado para proceder con el respectivo procedimiento disciplinario. Una segunda advertencia será reportada para el inicio del procedimiento disciplinario correspondiente.
11. Se recomienda al estudiante estar atento/a a los datos de su evaluación. La consignación de datos que no correspondan a su evaluación será considerado indicio concluyente de copia.

UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

SÍLABO DEL CURSO

1. ASIGNATURA

CS1111 – Programación I

2. DATOS GENERALES

2.1 Ciclo: NIVEL 1,NIVEL 2
2.2 Créditos: 4
2.3 Condición: Obligatorio
2.4 Idioma de dictado: Español
2.5 Requisitos: CC1102 - Pensamiento Matemático

3. INTRODUCCIÓN AL CURSO

Este curso de Programación I, de naturaleza teórica y práctica, aborda fundamentalmente los conceptos y técnicas esenciales para el desarrollo de habilidades de programación. Desde la comprensión de algoritmos hasta la implementación de estos usando estructuras de datos como listas y diccionarios. Se explora la resolución de problemas mediante la aplicación de estructuras de control, funciones, y recursividad. Además, se introduce el manejo de archivos y se profundiza en la complejidad algorítmica, evaluando el rendimiento de los algoritmos.

4. OBJETIVOS

- Sesión 1 Discutir la importancia de los algoritmos en el proceso de solución de un problema.
- Sesión 2 Identificar, describir y escribir programas que usan tipos de datos primitivos.
- Sesión 3 Escoger estructuras de condición adecuadas para una tarea de programación dada.
- Sesión 4 Escoger estructuras de repetición adecuadas para una tarea de programación dada.
- Sesión 5 Implementar un algoritmo de divide y vencerás para resolver un problema usando funciones.
- Sesión 6 Identificar y describir el uso de cadenas de texto y escribe programas que usan cadenas de texto.
- Sesión 7 Analizar y explicar el comportamiento de programas simples que involucran estructuras fundamentales de programación, variables, expresiones, asig- naciones, E/S, estructuras de control, funciones, paso de parámetros.
- Sesión 8 Escribir un programa que usa E/S de archivos para brindar persistencia a través de ejecuciones múltiples.

- Sesión 9 Describir el concepto de recursividad y da ejemplos de su uso. Identifica el caso base y el caso general de un problema basado en recursividad y escribe programas basados en funciones recursivas.
- Sesión 10 Diseñar, implementar, probar, y depurar un programa que usa estructuras de datos como arreglos y listas.
- Sesión 11 Diseñar, implementar, probar, y depurar un programa que usa estructuras de datos como diccionarios o tablas de hash.
- Sesión 12 Usar la notación formal Big O para dar límites de casos esperados en el tiempo de complejidad de los algoritmos. Explica a que se refiere con “mejor”, “esperado” y “peor” caso de comportamiento de un algoritmo
- Sesión 13 Implementar algoritmos de ordenamiento y explicar las diferencias en sus tiempos de complejidad.
- Sesión 14 Implementar algoritmos de búsqueda simple y explicar las diferencias en sus tiempos de complejidad.

5. COMPETENCIAS Y CRITERIOS DE DESEMPEÑO

Competencias Especificas - NEGOCIOS

- Estructura, analiza y relaciona situaciones de negocio, tomando decisiones acertadas a partir del entendimiento del consumidor, los mercados, los procesos, las organizaciones y las finanzas.
- Comprende las principales tendencias en tecnología y su potencial para los negocios, identificando formas de utilizarlas para resolver problemas, con creatividad y pensamiento crítico.

Competencias Especificas ABET - INGENIERIA

- La capacidad de identificar, formular y resolver problemas complejos de ingeniería mediante la aplicación de principios de ingeniería, ciencias y matemáticas.

Competencias Especificas ABET - COMPUTACION

- Analizar un problema computacional complejo y aplicar principios de computación y otras disciplinas relevantes para identificar soluciones.
- Diseñar, implementar y evaluar una solución computacional para satisfacer un conjunto determinado de requerimientos computacionales en el contexto de la disciplina del programa.

6. RESULTADOS DE APRENDIZAJE

- Identificar las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos de un algoritmo como “mejor”, “esperado” y “peor” caso.
- Identificar las propiedades de un programa, las herramientas necesarias para su implementación y los recursos requeridos para su ejecución.

- Explicar el comportamiento de programas simples que involucran estructuras fundamentales de programación como variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros y recursividad.
- Identificar las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos de un algoritmo como “mejor”, “esperado” y “peor” caso.
- Identificar las propiedades de un programa, las herramientas necesarias para su implementación y los recursos requeridos para su ejecución.
- Explicar el comportamiento de programas simples que involucran estructuras fundamentales de programación como variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros y recursividad.
- Identificar las características de data y/o otras condiciones o suposiciones que lleven a diferentes comportamientos de un algoritmo como “mejor”, “esperado” y “peor” caso.
- Identificar las propiedades de un programa, las herramientas necesarias para su implementación y los recursos requeridos para su ejecución.
- Explicar el comportamiento de programas simples que involucran estructuras fundamentales de programación como variables, expresiones, asignaciones, E/S, estructuras de control, funciones, paso de parámetros y recursividad.

7. TEMAS

1. Algoritmos

1.1. ¿Qué es un algoritmo?
1.2. ¿Cómo se representa?

2. Introducción a Python

2.1. ¿Qué es Python?
2.2. ¿Qué es un intérprete?
2.3. ¿Qué es un IDE?

3. Entradas y Salidas

3.1. Tipos de datos básicos
3.2. Variables
3.3. Operadores aritméticos y lógicos

4. Estructuras de Control

4.1. Estructuras de control selectivas
4.2. Estructuras de control repetitivas
4.3. Estructuras de control repetitivas anidadas

5. Strings

5.1. ¿Qué es un string?
5.2. Funciones básicas de strings
5.3. Índices y operadores de strings
5.4. Iteración en un string

6. Funciones

- 6.1. ¿Qué es una función?
- 6.2. Parámetros por valor y referencia
- 6.3. Librerías
- 6.4. Alcance de una variable
- 6.5. Parámetros por defecto

7. Listas

- 7.1. ¿Qué es una lista?
- 7.2. Índices y operadores en una lista
- 7.3. Iteración en una lista
- 7.4. Matrices
- 7.5. Listas por comprensión

8. Diccionarios

- 8.1. ¿Qué es un diccionario?
- 8.2. Dupla: Llave - Valor

9. Archivos

- 9.1. Lectura de archivos
- 9.2. Escritura de archivos

10. Complejidad algorítmica

- 10.1. Complejidad en tiempo
- 10.2. Complejidad en espacio

11. Recursión

- 11.1. ¿Qué es recursión?
- 11.2. Cola de llamadas
- 11.3. Backtracking

12. Ordenamiento

- 12.1. Ordenamiento por selección (selection sort)
- 12.2. Ordenamiento por inserción (insertion sort)
- 12.3. Ordenamiento rápido (quicksort)

13. Búsqueda

- 13.1. Búsqueda Lineal
- 13.2. Búsqueda Binaria

8. PLAN DE TRABAJO

8.1 Metodología

El curso presenta una metodología de aprendizaje activo, basada en la resolución de problemas de una manera práctica con enfoque al conocimiento adquirido en sesiones teóricas.
Adicionalmente, existe aprendizaje a través de la ejecución de un proyecto, de esta manera, se anima y motiva al estudiante a que continúe con su proceso de aprendizaje.

8.2 Sesiones de teoría

Las sesiones de teoría se llevan a cabo en clases magistrales donde se realizarán actividades que propicien un aprendizaje activo, con dinámicas que permitan a los estudiantes interiorizar los conceptos.

8.3 Sesiones de práctica (laboratorio o taller)

Para verificar que los alumnos hayan alcanzado el logro planteado para cada una de las unidadesde aprendizaje, realizarán actividades que les permita aplicar los conocimientos adquiridosdurante las sesiones de teoría y se les propondrá retos que permitan evaluar el desempeñode los alumnos.

9. SISTEMA DE EVALUACIÓN

El curso consta de los siguientes espacios de evaluación:

	Teoría
Evaluación	**EVALUACIÓN** La ponderación de la evaluación se hará si ambas partes (teoría y laboratorio) están aprobadas. **TEORÍA (40%)** Examen E1 (10%) Examen E2 (10%) Tarea C1 (5%) Tarea C2 (5%)

	Tarea C3 (5%) Tarea C4 (5%) **LABORATORIO (60%)** Evaluación Continua C1 (4%) Evaluación Continua C2 (4%) Práctica Calificada PC1 (9%) Práctica Calificada PC2 (9%) Práctica Calificada PC3 (9%) Práctica Calificada PC4 (9%) Proyecto P1 (8%) Proyecto P2 (8%)
	100%

10. REFERENCIAS BIBLIOGRÁFICAS

[1] BROOKSHEAR, J. G., SMITH, D., AND BRYLOW, D. Computer science: an overview.

[2] FERREIRA FILHO, W. Computer Science Distilled: Learn the Art of Solving Computational Problems. Code Energy LLC, 2017.

[3] GUTTAG, J. V. Introduction to computation and programming using Python: With applicationto understanding data. MIT Press, 2016.