LENGUAJE DE PROGRAMACIÓN I

REFLEXIÓN

“La educación no cambia el mundo, cambia a las personas que van a cambiar el mundo”.

Paulo Freires

TEMARIO

- Presentación del docente.
- Sílabo.
- Normativas de curso.
- Excepciones y funciones en Python.

PRESENTACIÓN

MAG. DELY MARYSHECK LAZO BARREDA – dlazo@ucsm.edu.pe

LENGUAJE DE PROGRAMACIÓN I

SÍLABO

Sílabo de la asignatura

MAG. DELY LAZO BARREDA

UNIVERSIDAD CATÓLICA DE SANTA MARÍA AREQUIPA

FACULTAD: CIENCIAS E INGENIERIAS FISICAS Y FORMALES

DEPARTAMENTO ACADÉMICO: CIENCIAS E INGENIERIAS FISICAS Y FORMALES

ESCUELA PROFESIONAL: INGENIERIA DE SISTEMAS

SÍLABO DE ASIGNATURA

1. IDENTIFICACIÓN ACADÉMICA

Nombre de la asignatura: LENGUAJES DE PROGRAMACIÓN I

Código de la asignatura: 7102393

Semestre Académico en que se desarrolla: 2

CRÉDITOS	HORAS SEMANALES				HORAS SEMESTRALES		
	Horas Teóricas	Práctica de Aula	Jefe de Prácticas	Horas Virtuales	Horas Teóricas	Horas Prácticas	Horas Virtuales
4	2	0	4	0	36	72	0

El desarrollo de las actividades académicas se distribuye en tres fases. Cada semestre académico comprende dieciocho semanas. (Resolución N° 6199-CU-2016)

2. SUMILLA

Este curso de Lenguaje de Programación I es de la modalidad teórico práctica, es un curso del área de especialidad, tendiente a la mejora de la capacidad para identificar, comprender y mejorar la construcción de soluciones en sistemas organizacionales, se incide en la mejora del razonamiento lógico del estudiante que le permitan hacer programas más completos en lenguaje Python. Considerando que ya conoce el concepto de las estructuras de programación, se inicia con el tema de funciones y módulos, luego se continúa el manejo de clases y objetos. La segunda fase trata el almacenamiento de datos, inicia el manejo de archivos y luego las bases de datos. En la tercera fase el tema es el lenguaje C y sus semejanzas con el Python en las estructuras de programación. Todos estos contenidos y el desarrollo del curso deben mejorar la capacidad de liderazgo y una actitud positiva del estudiante

3. COMPETENCIAS DEL PERFIL DE EGRESO

- [x] ÉTICA
- [x] EXPERIMENTACIÓN
- [x] APRENDIZAJE PERMANENTE
- [x] USO DE HERRAMIENTAS MODERNAS

4. COMPETENCIAS ESPECÍFICAS DE LA ASIGNATURA

- [x] Compara e introduce herramientas de programación utilizando funciones y tecnología orientada a objetos en forma eficaz
- [x] Analiza y aplica los principios de la programación utilizando archivos y bases de datos para lograr optimizar el uso de la computadora al solucionar algoritmos de almacenamiento de los datos.
- [x] Concluye y afina los criterios más importantes sobre el uso del python y el C al relacionarlos con conceptos avanzados de computación

5. RESULTADOS DE APRENDIZAJE

	Tipo	Aprendizaje
Primera Fase	Conocimiento	El estudiante muestra capacidad para construir programas en lenguaje Python utilizando funciones y POO
	Desempeño	El estudiante muestra capacidad para construir programas con funciones
	Producto	El estudiante muestra competencia para hacer informes de las prácticas
Segunda Fase	Conocimiento	El estudiante muestra capacidad para estructurar programas que utilicen medios de almacenamiento permanente
	Desempeño	El estudiante muestra capacidad para diseñar archivos y bases de datos
	Producto	El estudiante muestra capacidad para hacer informes sobre el almacenamiento permanente
Tercera Fase	Conocimiento	El estudiante muestra capacidad para estructurar programas en lenguaje C
	Desempeño	El estudiante muestra capacidad para diseñar programas en C
	Producto	El estudiante muestra competencia para hacer informes con programas en C

6. CONTENIDOS

PRIMERA FASE

- Funciones y POO
  - Funciones y Módulos en Python
    - Definición de funciones
    - Paso de Parámetros
    - Ámbito de las variables
    - Que son los Módulos
    - Funcionamiento de la importación
  - Orientación a Objetos en Python
    - La programación orientada a Objetos
    - Clases y Objetos
    - Python y los Objetos
    - Atributos y Métodos
    - Tipos de variables
    - Otros conceptos de la orientación a Objetos

SEGUNDA FASE

- Almacenamiento de Datos
  - Almacenamiento de Datos. Archivos
    - Operaciones Básicas
    - Tipos de Ficheros
    - Manejos especiales
  - Bases de Datos
    - Del Bit a las Bases de Datos
    - Registros y Tablas
    - Bases de Datos Relacionales
    - Motor de Base de Datos SQLite
    - Operaciones con Bases de Datos
    - Consultas

TERCERA FASE

- Programación en C
  - Relaciones entre Python y el lenguaje C
  - Características del lenguaje C
  - Comparación de Estructuras de Python y C
  - Estructuras de decisión y repetición
  - Funciones en C
  - Arreglos en C

7. ESTRATEGIAS DIDÁCTICAS.

	Nro	Estrategia
Primera Fase	1	Clase Magistral
	2	Estudio de Casos
	3	Trabajo en Equipo
	4	Lluvia de Ideas
	5	Videos Tutoriales
	6	Aprendizaje Basado en Problemas
	7	Experimentación
Segunda Fase	1	Clase Magistral
	2	Estudio de Casos
	3	Trabajo en Equipo
	4	Lluvia de Ideas
	5	Videos Tutoriales
	6	Aprendizaje Basado en Problemas
	7	Experimentación
Tercera Fase	1	Clase Magistral
	2	Estudio de Casos
	3	Trabajo en Equipo
	4	Lluvia de Ideas
	5	Videos Tutoriales
	6	Aprendizaje Basado en Problemas
	7	Experimentación

8. EVALUACIÓN DE LOS APRENDIZAJES (criterios de evaluación)

	Tipo	Evaluación	Se califica con:
Primera Fase	Evidencia del Conocimiento	Exámen escrito	
		Participación oral	
	Evidencia del Producto	Monografías	Lista de cotejo
	Evidencia del Desempeño	Rúbricas	
Segunda Fase	Evidencia del Conocimiento	Exámen escrito	
		Participación oral	
	Evidencia del Producto	Monografías	Lista de cotejo
	Evidencia del Desempeño	Rúbricas	
Tercera Fase	Evidencia del Conocimiento	Exámen escrito	
		Participación oral	
	Evidencia del Producto	Monografías	Lista de cotejo
	Evidencia del Desempeño	Rúbricas	

10. BIBLIOGRAFÍA

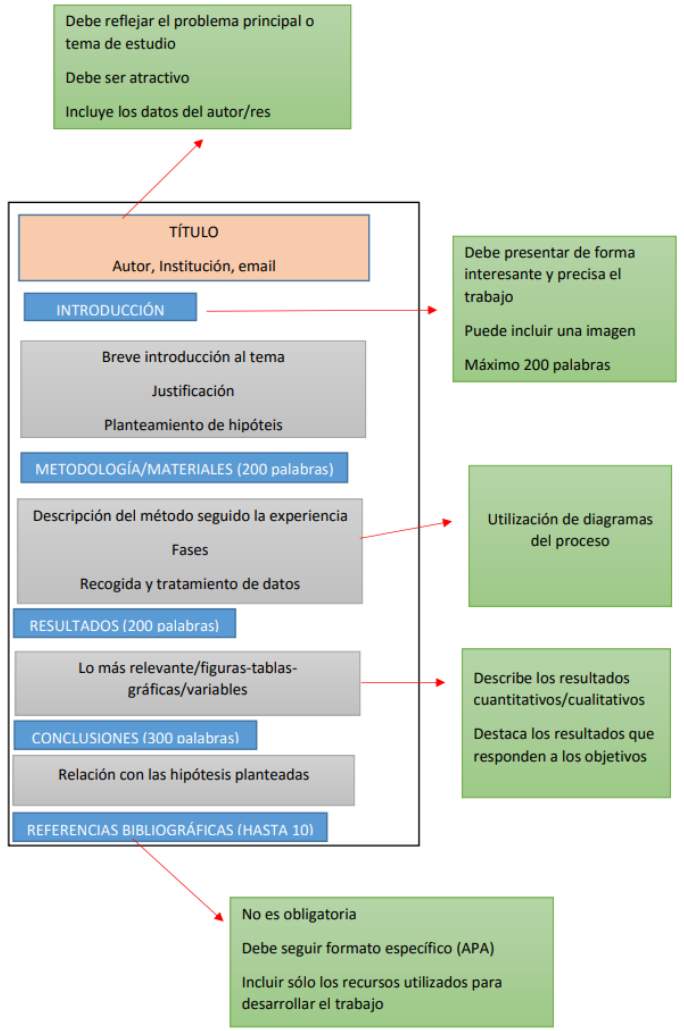
☑	Jorge Nolasco Valenzuela,Javier Gamboa Cruzado,Luz Nolasco Valenzuela (2020-01-01 00:00:00) Fundamentos de Programación con Python 3 MACRO>9786123046842	Básica
☑	BUTTU, MARCO (2016) EL GRAN LIBRO DE PYTHON - ED. 1 / 2016 MARCOMBO>005.133.BUTT.00	Básica
☑	DEITEL, HARVEY M. DEITEL, PAUL (2009) C++ COMO PROGRAMAR 6ta. Edic. INC. CD ROM PEARSON EDUCACION>005.133.DEIT.06	Clásica

DOCENTES Y JEFES DE PRÁCTICA RESPONSABLES DE LA ASIGNATURA		
Nombre y Apellidos	**Categoría**	**Correo electrónico**
ZUÑIGA CARNERO, MANUEL MARIANO	Principal	mzunigac@ucsm.edu.pe
LAZO BARREDA, DELY MARYSHECK	Docente Contratado	dlazo@ucsm.edu.pe
IQUIRA BECERRA, DIEGO ALONSO	Jefe Practicas Contratado	diquira@ucsm.edu.pe

TRABAJOS DE INVESTIGACIÓN FORMATIVA Y RESPONSABILIDAD SOCIAL

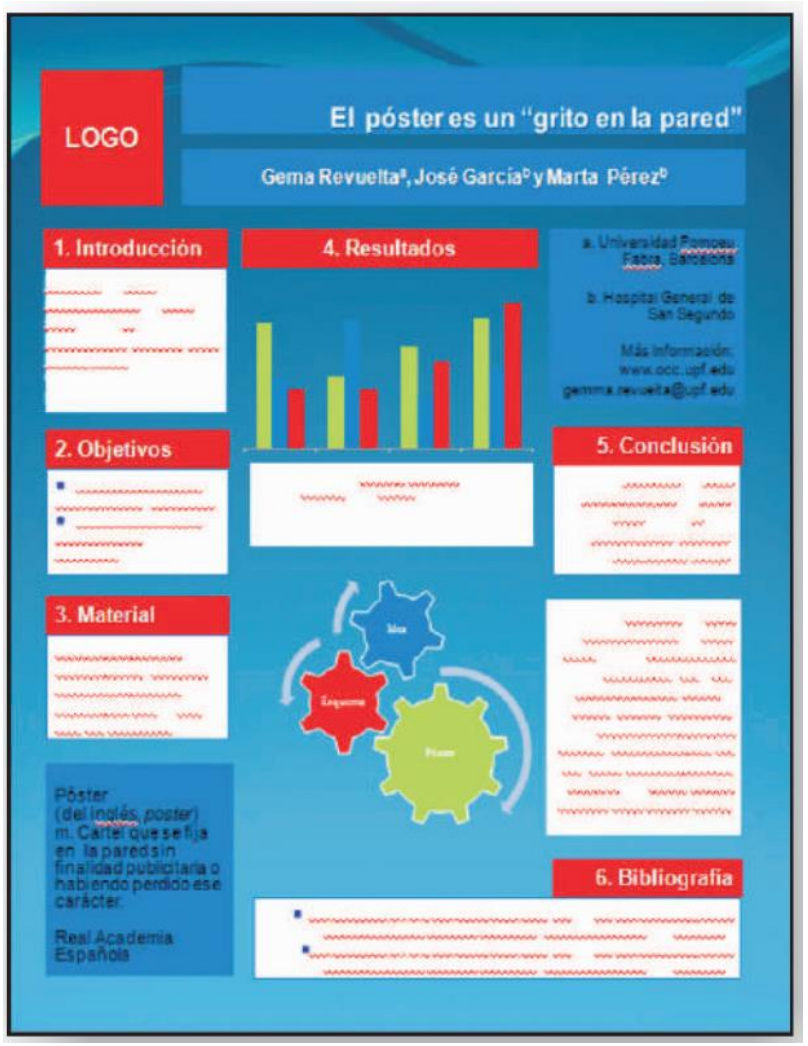
INVESTIGACIÓN FORMATIVA

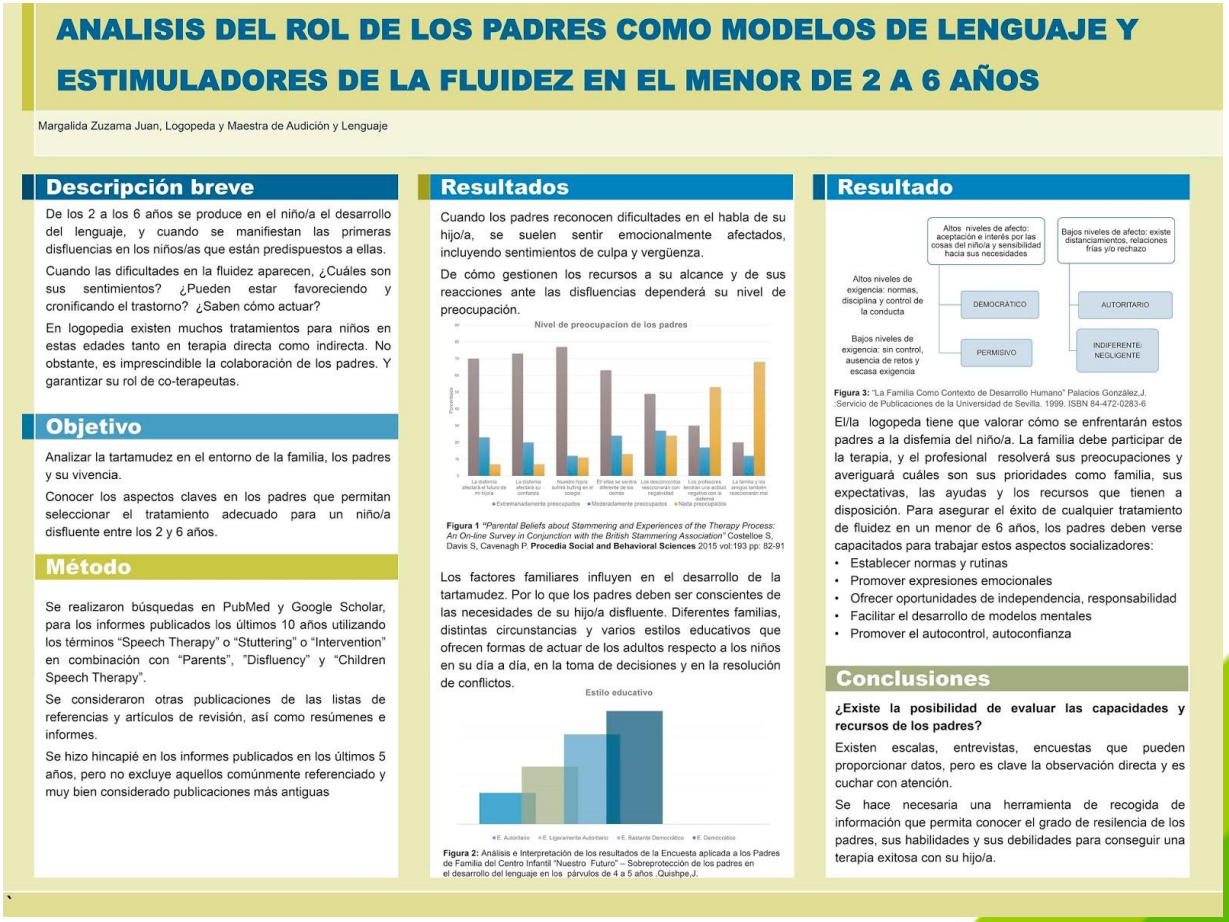
Estructura del póster científico



- Realizar en grupos de 3 participantes como máximo un póster científico tema relacionado con nuestra carrera de Ingeniería de sistemas. Fecha de entrega fin de primera fase.

INVESTIGACIÓN FORMATIVA





RESPONSABILIDAD SOCIAL

- Realizar en grupos de 3 participantes como máximo una charla o conferencia mínimo 120 minutos de lo aprendido de Python en una institución educativa primaria o secundaria. Fecha de entrega fin de tercera fase.

CRONOGRAMA Y CALIFICATIVO DE LA ASIGNATURA

FASE	DESDE	HASTA
I	14/08/2023	23/09/2023
II	25/09/2023	04/11/2023
III	06/11/2023	09/12/2023

ITEM	PUNTOS
Asistencia	1
Participación	3
Práctica Final	4
Ejercicios individuales	5
Informes grupales	5
Lectura	2
TOTAL	20

LECTURA	FASE	ENLACE
La vaca	1	http://courseware.url.edu.gt/Facultades/Facultad%20de%20Teolog%C3%ADa/Magis%20Landivariano%20Material/Libros/Libro%20La%20VACA.pdf
El hombre mas rico de Babilonia	2	https://irp-cdn.multiscreensite.com/3af19a39/files/uploaded/EL%20HOMBRE%20MAS%20RICO%20DE%20BABILONIA%20-%20GEORGE%20S.%20CLASON%20-%2086%20PAGINAS%282%29.pdf
Padre rico, padre pobre	3	https://www.economicas.unsa.edu.ar/afinan/informacion_general/book/padre-rico-padre-pobre.pdf

FORMATOS ENTREGA DE TRABAJOS

ENSAYO (FORMATO SIMPLE)

- Realizar en forma individual un ensayo sobre la lectura asignada.
- El ensayo debe consignar las siguientes partes:
  - Título.
  - Introducción: Responde a la pregunta general: ¿de qué trata este ensayo?, ocupa un 15% del texto.
  - Desarrollo: Es la exposición detallada de tus pensamientos sobre la lectura efectuada. De todas las partes de un ensayo, esta es la más creativa. El desarrollo se compone de ideas, reflexiones y argumentos sobre el objeto de análisis y ocupa casi todo el texto del ensayo (más o menos el 75 %). Mencione como puede aplicar los conocimientos adquiridos de la lectura en su vida profesional y personal.
  - Conclusiones: consiste en unas pocas ideas que ponen fin al tema. Igual que la introducción, ocupa un 15% del texto.
  - Anexos, bibliografía (opcional).

PRESENTACIÓN DE TRABAJOS

- Los trabajos que se presenten en el curso deberán contar con una **portada** o carátula, **donde se incluya los nombres de todos los integrantes**.
- Se deben presentar únicamente documentos en **formato PDF**, en caso de tener que agregar códigos fuente, audio, videos u otros estos serán por enlaces con los permisos respectivos.

FORMATO SIMPLE

UNIVERSIDAD CATÓLICA DE SANTA MARÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

<<NOMBRE ASIGNATURA>>
<<NOMBRE ACTIVIDAD/TRABAJO>>

CÓDIGO/DNI	APELLIDOS Y NOMBRES	FECHA

Si el trabajo es grupal, se colocará a los integrantes en orden alfabético por el apellido paterno.

<<CUERPO DE LA ACTIVIDAD/TRABAJO>>

Deberá considerar las especificaciones hechas por el docente del curso.

<<REFERENCIAS>>

El estudiante debe listar todas las fuentes utilizadas para desarrollar la actividad/trabajo.

FORMATO INFORME TÉCNICO

ESQUEMA GENERAL DEL INFORME

(Se recomienda leer/revisar un artículo científico redactado en estándar IEEE para comprender lo que se explica a continuación)

Título del proyecto
Nombre apropiado para el proyecto, debe describir en un máximo de tres líneas su trabajo.

Autores
Nombres de los encargados del proyecto, en orden alfabético por apellido.

Resumen ejecutivo
Esta sección se elabora una vez que se finaliza el informe, y como dice su nombre, resume lo que es entrega en ese documento.

Introducción
Toda introducción debe contener al menos 4 partes, no necesariamente en 4 párrafos ni tampoco separadas con títulos o subtítulos. Estas partes son:

- Contexto: Explicación general del tema, que lleve a entender el problema específico que se trata en el documento. Es una introducción al tema, la amplitud de esta subsección dependerá de la complejidad del tema que están tratando. Esta subsección debe estar bien sustentada con citas y sus respectivas referencias al final del documento.
- Problema o necesidad: Breve explicación del problema o necesidad que se pretende abordar en el documento. De ser necesario, también debe contener citas.
- Tarea: Resumen de lo que se ha hecho en el documento; esta subsección es más amplia que el resumen ejecutivo.
- Estructura del documento: Esta subsección si es un párrafo aparte, y muestra la estructura del documento por sus títulos principales.

Cuerpo del informe
Contenido de cada entrega, el cual será detallado y explicado por el docente de la asignatura. Debe incluir todas las citas necesarias, para sustentar su trabajo.
El cuerpo del informe se puede escribir en un solo título y con sus subtítulos correspondientes; o, en diferentes títulos.

Conclusiones
Listado de ideas, hallazgos o descubrimientos relevantes del informe, que se derivan del mismo trabajo.

Referencias
Listado de documentos utilizados para sustentar su trabajo, deben seguir la estructura que define el estándar IEEE.

Anexos/Apéndices
Listado de documentos y/o textos que complementan el trabajo.
Cada anexo inicia en una página nueva y debe tener un título.

Considerando el estándar IEEE, elabore el informe tomando en cuenta la siguiente estructura, como mínimo.

Recuerde que el estándar IEEE no especifica que debe hacerse a dos columnas.

Ejemplo de un artículo IEEE

NOMBRAMOS A SU REPRESENTANTE

1. EL DELEGADO TENDRA POR FUNCIÓN SER EL COMUNICADOR ENTRE LOS ESTUDIANTES Y DOCENTE.
2. LLEVAR EL CONTROL DE LAS PARTICIPACIONES.

REGLAS A SEGUIR

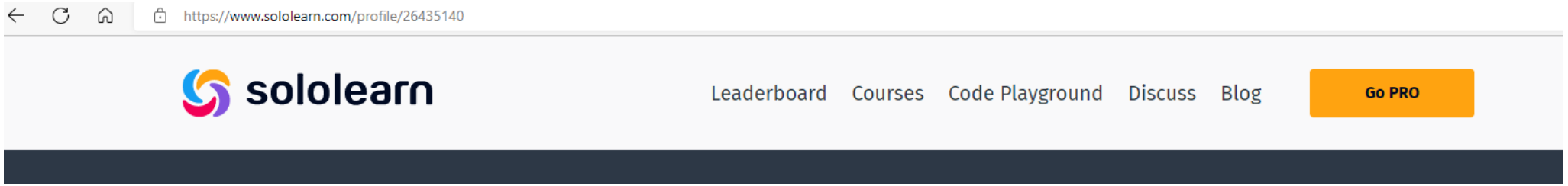
1. Mientras una persona toma la palabra los demás prestamos atención guardando el silencio respectivo.
2. La calificación de la práctica se evalúa durante las dos horas de trabajo.
3. La fecha de la entrega de los informes es hasta el mismo día de la práctica hasta las 23:59 horas.
4. La asistencia a prácticas es obligatoria (desde el inicio hasta el final), se considera una tolerancia máxima de 10 minutos sobre la hora de ingreso fijada.
5. Los estudiantes que no asistan por algún tipo de urgencia médica u otro de fuerza mayor, deberán solicitar el permiso correspondiente, en cuyo caso se les justificará la inasistencia pero no tendrán derecho a los puntos por asistencia.
6. El estudiante que registre 5 inasistencias injustificadas durante el semestre será retirado de prácticas, obteniendo una notal final de cero.
7. Esta prohibido comer o beber en los laboratorios.
8. Las prácticas finalizarán 10 minutos antes para dar espacio a la salida e ingreso del nuevo grupo al laboratorio.
9. Los informes grupales se pueden desarrollar como máximo 3 participante por grupo y sólo uno sube la evidencia.

PREGUNTAS O DUDAS

SESIÓN INTRODUCTORIA

REPASO

Práctica



Ingresar a la web www.sololearn.com y practicar los siguientes puntos:

- Getting started with Python.
- Going deeper with Python.
- Working with data.
- Control flow.


Introduction to Python
Python is a popular, easy-to-learn, and very powerful programming language, which is used in software and web development, data science, machine learning, and many other fields. In this course, we'll cover the basic concepts of Python, as well as build real-life projects and solve different coding challenges. Introduction to Python requires no prior programming experience, so let's dive right in!
Getting started with Python
Going Deeper with Python
Working with Data
Control Flow
Lesson
Control Flow
XP +10

Ecosistema de Python

1. Instalar Python (Framework)
   a. https://www.python.org
   b. Intérprete (python.exe)
   c. Librerías Estándar
      i. Acceso a archivos y directorios.
      ii. Servicios de procesamiento de textos.
      iii. Persistencia de datos.
2. Recursos Externos más usados en Python.
   a. Librerías científicas:
      i. Cálculo y/o procesamiento datos.
      ii. Numpy, Matplotlib, Scipy
      iii. Pandas (Dataframe).
   b. Librerías de Acceso a Datos.
      i. PyOdbc.
   c. Entorno de Desarrollo Integrado
      i. VSCode. (https://code.visualstudio.com)
      ii. PyCharms.
      iii. Spyder
   d. Entorno de trabajo Web Colaborativo
      i. Jupyter Lab.
      ii. Google Colab.
      iii. Jupyter Notebook

LENGUAJE DE PROGRAMACIÓN I

SESIÓN Nº 1

FUNCIONES EN PYTHON

MAG. DELY LAZO BARREDA

Objetivos

- Aplicar el uso de excepciones en Python.
- Conocer las funciones en Python.
- Aprender a diseñar funciones en Python.
- Apreciar las características y ventajas provistas por las funciones para resolver problemas de programación (tipos de valores, número indefinido de parámetros, etc.).

Temas a tratar

- Excepciones.
- Introducción a Funciones.
- ¿Qué es una función con argumentos?
- Funciones con parámetros por defecto.
- ¿Qué es una función con una cantidad variable de parámetros?
- Funciones con una cantidad variable de parámetros.

EXCEPCIONES

- Los **errores** que se presentan en ejecución son llamadas excepciones.
- Durante la ejecución de un programa, si dentro de una función aparece una excepción y la función no la maneja, la **excepción se propaga hacia la función que la invocó**.
- Es una forma de controlar el comportamiento de un programa cuando se produce un error.

EXCEPCIONES EJEMPLO 1

```
num1=5
num2=6
try:
    num1/num2
except:
    print("Error")
else:
    print("Todo bien")
finally:
    print("Finalizo")])
```

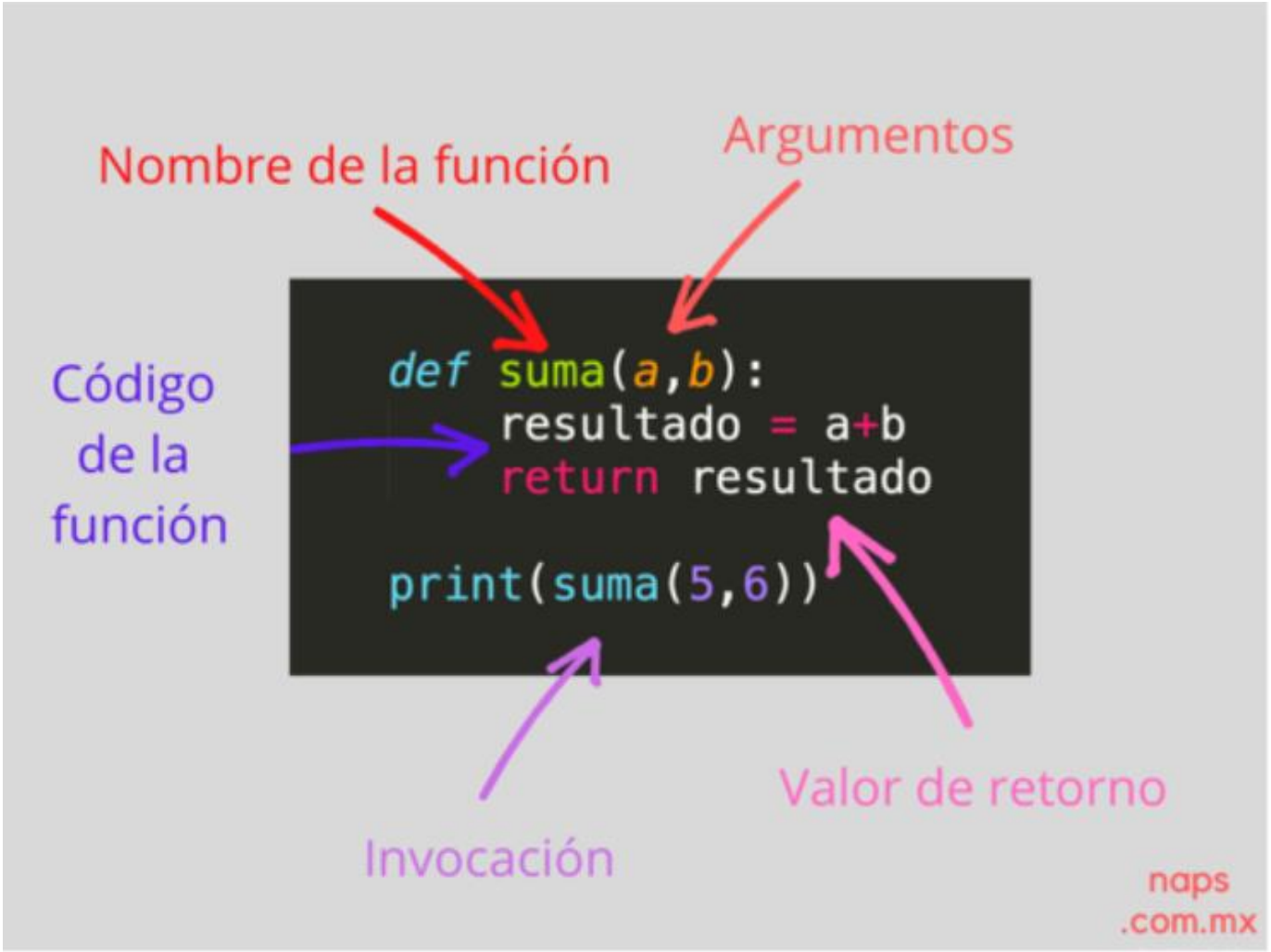
```
# Bloque try: Bloque que permite probar la búsqueda de errores.
# Bloque except: Bloque que permite manejar los errores.
# Bloque else: Bloque que se ejecuta cuando no hay ningún error.
# Bloque finally: Bloque que se ejecuta cuando finaliza la excepción.
```

EXCEPCIONES EJEMPLO 2

```
def validar_numero_entero():
    while True:
        try:
            valor = int(input("Ingrese un número: "))
            if valor != 0:
                return valor
                break
            else:
                print("El valor ingresado no puede ser 0.")
        except:
            print("El valor ingresado no es un número entero.")

numero = validar_numero_entero()
print(f"El numero ingresado es {numero}")
```

FUNCIONES EJEMPLO 3



FUNCIONES EJEMPLO 4

```
#Sintaxis de una función
#Definición de una función
def nombre_funcion(argumento1, argumento2):
    # Cuerpo de la función
    resultado = argumento1 + argumento2
    return resultado
```

FUNCIONES EJEMPLO 5

- La función, no se ejecuta hasta que no sea invocada. Para invocar una función, se la llama por su nombre:

```
#Invocación de una función
def mi_funcion():
    print("Hola Mundo")


mi_funcion()
```

```
Hola Mundo
```

FUNCIONES EJEMPLO 6

- Cuando una función retorne datos, éstos, se pueden asignar a una variable.

```
#Retorno de valor de una función
def funcion():
    return "Hola Mundo"

saludo = funcion()
print(saludo)
```

```
Hola Mundo
```

FUNCIONES EJEMPLO 7

- Definimos la función impresion(), que muestra un texto determinado repetidas veces y enmarcado por un símbolo a elegir. Su definición es la siguiente:

```
#Función con argumentos
def impresion(veces, simbolo):
    print(simbolo*28)
    for n in range(veces):
        print(simbolo,"Ejemplo de impresion ...",simbolo)
    print(simbolo*28)
impresion(3,'*')
```

```
****************************
* Ejemplo de impresion ... *
* Ejemplo de impresion ... *
* Ejemplo de impresion ... *
****************************
```

FUNCIONES EJEMPLO 8

- Definimos la función impresion() con el parámetro por defecto.

```
#Función con parámetros por defecto
def impresion(veces, simbolo='+'):
    print(simbolo*28)
    for n in range(veces):
        print(simbolo,"Ejemplo de impresion ...",simbolo)
    print(simbolo*28)

impresion(3)
impresion(3,"$")
```

```
++++++++++++++++++++++++++++
+ Ejemplo de impresion ... +
+ Ejemplo de impresion ... +
+ Ejemplo de impresion ... +
++++++++++++++++++++++++++++
$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ Ejemplo de impresion ... $
$ Ejemplo de impresion ... $
$ Ejemplo de impresion ... $
$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

FUNCIONES EJEMPLO 8

- Es importante tener en cuenta que los **argumentos por defecto deben figurar siempre después de los requeridos**, y nunca antes.

```
    def impresion(simbolo='+',veces):
                                 ^^^^^
SyntaxError: non-default argument follows default argument
```

FUNCIONES EJEMPLO 9

- Función parámetros con valores por defecto y otros sin valor por defecto

```
#Función con tres parámetros, dos de ellos con valores por defecto
def mifuncion(a,b=2,c=3):
    print(a+b+c)

mifuncion(4,5,6)
mifuncion(6,4)
mifuncion(5)
mifuncion(a=10,c=3)
```

```
15
13
10
15
```

FUNCIONES EJEMPLO 10

- Python nos permite crear funciones que acepten un número indefinido de parámetros. Los operadores * y ** son los que se utilizan para esta funcionalidad.

```
#Función con una cantidad variable de
parámetros con una invocación
def elementos(*args):
   print("Elementos: ", args)


elementos('a', 'b', 'c', 'd')
```

```
Elementos:  ('a', 'b', 'c', 'd')
```

FUNCIONES EJEMPLO 11

```
#Función con una cantidad variable de parámetros con n invocaciones
def myFunction(*items):
    for item in items:
        print(item)
print("Primera invocacion: ")
myFunction()
print("Segunda invocacion: ")
myFunction(1, 2, 3)
print("Tercera invocacion: ")
myFunction('a', 'b', 'c', 'd', 2, 3)
tupla = ('a', 1, 'b', 2, 'c', 3)
print("Cuarta invocacion: ")
myFunction(tupla)
```

```
Primera invocacion:
Segunda invocacion:
1
2
3
Tercera invocacion:
a
b
c
d
2
3
Cuarta invocacion:
('a', 1, 'b', 2, 'c', 3)
```

FUNCIONES EJEMPLO 12

- Función con n variables de parámetros haciendo uso de tupla y diccionario.

```
#Función con una cantidad variable de parámetros – tupla diccionario
def elementos(*args, **kwargs):
    print("Argumentos posicionales:", args)
    print("Argumentos con nombre:", kwargs)

elementos(valor1='a', valor2='b', valor3='c', valor4='d')
elementos('a', 'b', 'c', 'd')
```

```
Argumentos posicionales: ()
Argumentos con nombre: {'valor1': 'a', 'valor2': 'b', 'valor3': 'c', 'valor4': 'd'}
Argumentos posicionales: ('a', 'b', 'c', 'd')
Argumentos con nombre: {}
```

FUNCIONES EJEMPLO 13

- Función con cantidad variable de parámetros haciendo uso de diccionario.

```
#Función con una cantidad variable de parámetros - diccionario
def miFuncion(**argumentos):
    print(argumentos)
miFuncion(x = 1, y = 3)
miFuncion(x = 'a', y = 'b', z = 'c')
```

```
{'x': 1, 'y': 3}
{'x': 'a', 'y': 'b', 'z': 'c'}
```

FUNCIONES EJEMPLO 14

- Función con cantidad variable de parámetros y otros.

```
Función con una cantidad variable de parámetros y otros
def myFunction(nombre, edad, **datos):
    print("Nombre: ", nombre)
    print("Edad: ", edad)
    for clave, valor in datos.items():
        print(clave, valor)

myFunction("Cristina", 24, Pais = "Estados Unidos")
myFunction("Carlos Santana", 25, Celular = 123456789, Pais = "México")
myFunction("Carlos Santana", 25, Celular = 123456789, Pais = "México", Ciudad ='Rivera Maya')
```

```
Nombre:  Cristina
Edad:  24
Pais Estados Unidos
Nombre:  Carlos Santana
Edad:  25
Celular 123456789
Pais México
Nombre:  Carlos Santana
Edad:  25
Celular 123456789
Pais México
Ciudad Rivera Maya
```

FUNCIONES EJEMPLO 15

- Función suma con cantidad variable de parámetros y otros.

```
 #Función suma con una cantidad variable de parámetros y otros
def sumar(x1,x2,*xn):
    s=x1+x2
    for valor in xn:
        s=s+valor
    return s

print(sumar(1, 2))
print(sumar(1,2,3,4))
print(sumar(1,2,3,4,5,6,7,8,9,10))
```

```
3
10
55
```

FUNCIONES EJEMPLO 16

```
def valida_numero(texto):
    while True:
        try:
            numero=int(input(texto))
            return numero
        except ValueError:
            print("Solo debe ingresar valores numéricos")
        except:
            print("Error en ingreso. Intente nuevamente! ")

def valida_nombre(texto):
    while True:
        try:
            nombre=input(texto)
            if nombre!="" and nombre.isalpha() and nombre[0].isupper():
                return nombre
            else:
                print("El nombre no debe ser vacío y la primera letra en
mayúsculas")
        except ValueError:
            print("Ingrese solo texto")
        except:
            print("Error en ingreso. Intente nuevamente! ")

def saludo():
    print("Bienvenido al curso LP1")
```

```
def saludo_argumento(nombre):
    print("Bienvenido al curso LP1 " + str(nombre))

def suma(a,b):
    resultado= a + b
    return resultado

def suma_valores(*elementos):
    resultado=0
    for x in elementos:
        resultado += x
    return resultado

def imprimir(**diccionario):
    for x,y in diccionario.items():
        print(x,y)

def menu():
    print("##########OPCIONES#########")
    print("1. Saludar")
    print("2. Saludar con argumentos")
    print("3. Sumar dos valores")
    print("4. Sumar n valores")
    print("5. Imprimir clave y valor")
    print("6. Salir")
    opcion=int(input("Elija opcion: "))
    return opcion
```

FUNCIONES EJEMPLO 16

```
while True:
    opc=menu()
    if opc ==1:
        saludo()
    elif opc == 2:
        nombre=valida_nombre("Ingrese un  nombre: ")
        saludo_argumento(nombre)
    elif opc == 3:
        numero1=valida_numero("Ingrese Primer Numero")
        numero2=valida_numero("Ingrese Segundo Numero")
        print("El resultado de sumar {} y {} es {} ".format(numero1,numero2,suma(numero1,numero2)))
    elif opc == 4:
        print("La suma de (1,2,3) es {}".format(suma_valores(1,2,3)))
        print("La suma de (11,12,13,14,15) es {}".format(suma_valores(11,12,13,14,15)))
        print("La suma de (21,22,23,24) es {}".format(suma_valores(21,22,23,24)))
    elif opc == 5:
        imprimir(codigo=1,nombre="Dely", apellido="Lazo")
        imprimir(codigo=2,nombre="Juan", apellido="Gomez", dni=12345678,direccion="Yanahuara")
    elif opc == 6:
        print("Termino el programa ...")
        break
    else:
        print("Error en elegir la opcion ")
```

ACTIVIDADES

I. Configuración VSCode

1. Descargue el aplicativo del siguiente URL: https://code.visualstudio.com/
2. Instalar Python desde el siguiente URL: https://www.python.org/.
3. Instalar la extensión Python en VSCode:

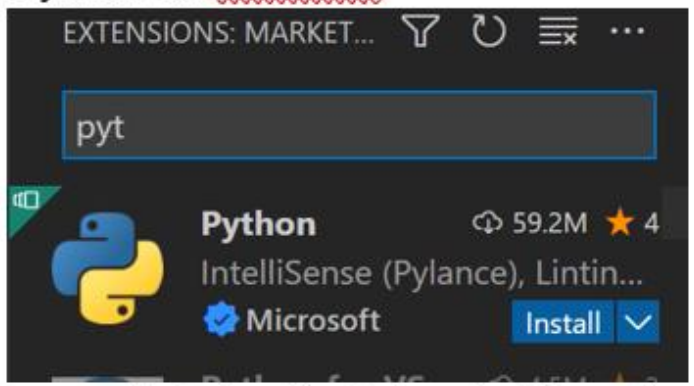


Imagen 13: Extensión Python de Microsoft

4. Instalar la extensión. run en VSCode.

Imagen 14: Extensión Code Runner

5. Crear una carpeta Lenguaje de Programación I desde el explorador. Luego seleccione dicha carpeta desde VSCode.

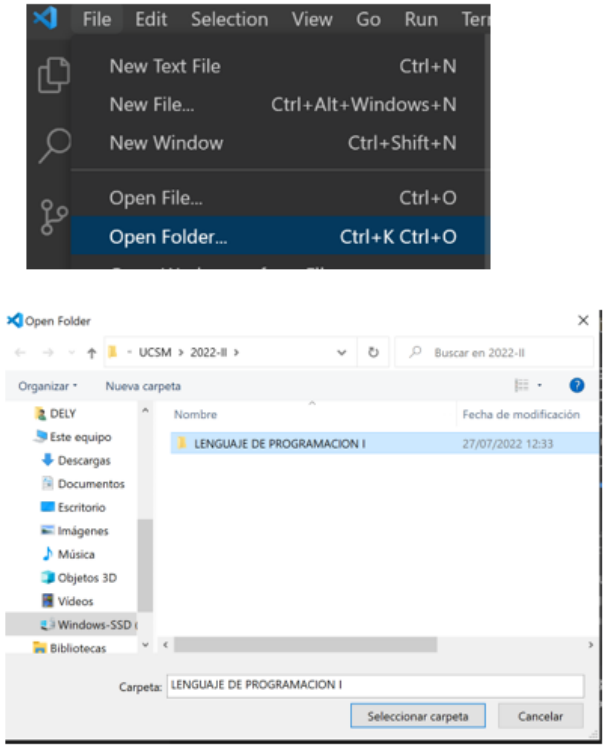


Imagen 15: Selección de carpeta en VSCode

6. Activar la opción Auto Save.

Imagen 16: Opción autoguardado

II. Funciones de Python

1. En el editor de Python escriba el siguiente código.

```
def saludo():
    print("Hola, esta es mi primera función")

saludo()
```

2. Guárdelo con el nombre de actividad1.py. Ejecute el programa con la opción Run.

Parámetros

3. En el editor de Python escriba el siguiente código.

```
def saludo_nombre(nombre):
    print(f"Hola, mi nombre es {nombre}.")

saludo_nombre("EPIS")
```

4. Guárdelo con el nombre de actividad2.py. Ejecute el programa con la opción Run.

Parámetros por omisión

5. En el editor de Python escriba el siguiente código.

```
def saludo_nombre(nombre,mensaje="Bienvenido al curso LP I"):
    print(f"{mensaje} estudiante {nombre}.")

saludo_nombre("EPIS")
saludo_nombre("María","Bienvenido Semestre Impar EPIS")
```

6. Guárdelo con el nombre de actividad3.py. Ejecute el programa con la opción Run.

Keywords con parámetros

7. En el editor de Python escriba el siguiente código.

```
def saludo(nombre, mensaje="Hola"):
    print (mensaje, nombre)

saludo(mensaje="Buen día", nombre="Juan")
```

8. Guárdelo con el nombre de actividad4.py. Ejecute el programa con la opción Run.

Parámetros arbitrarios

Para definir argumentos arbitrarios en una función, se antecede al parámetro un asterisco (*).

9. En el editor de Python escriba el siguiente código:

```
def suma(*args):
    total = 0
    for n in args:
        total += n
    return total

print(suma(2, 3, 5))
```

10. Guárdelo con el nombre de actividad5.py. Ejecute el programa con la opción Run.

Parámetros arbitrarios como pares de clave = valor

En estos casos, al nombre del parámetro deben precederlo dos asteriscos (**).

11. En el editor de Python escriba el siguiente código:

```
def suma(**kwargs):
    total = 0
    for value in kwargs.values():
        total += value
    return total

print(suma(a = 1, b = 3, c = 2))
```

12. Guárdelo con el nombre de actividad6.py. Ejecute el programa con la opción Run.

Desempaquetado de Parámetros

Puede ocurrir, además, una situación inversa a la anterior. Es decir, que la función espere una lista fija de parámetros, pero que éstos, en vez de estar disponibles de forma separada, se encuentren contenidos en una lista o tupla. En este caso, el signo asterisco (*) deberá preceder al nombre de la lista o tupla que es pasada como parámetro durante la llamada a la función.

13. En el editor de Python escriba el siguiente código:

```
def calcular(importe, descuento):
    return importe - (importe * descuento / 100)

datos = [1500, 10]
print(calcular(*datos))
```

14. Guárdelo con el nombre de actividad7.py. Ejecute el programa con la opción Run.

Llamadas de retorno

Es posible, llamar a una función dentro de otra, de forma fija y de la misma manera que se le llamaría, desde fuera de dicha función:

15. En el editor de Python escriba el siguiente código:

```
def funcion():
    return "Hola Mundo"

def llamada_de_retorno(func=""):
    """Llamada de retorno a nivel global"""
    return globals()[func]()

print (llamada_de_retorno("funcion"))

# Llamada de retorno a nivel local
nombre_de_la_funcion = "funcion"
print(locals()[nombre_de_la_funcion]())
```

16. Guárdelo con el nombre de actividad8.py. Ejecute el programa con la opción Run.

Llamadas recursivas

Python admite las llamadas recursivas, permitiendo a una función, llamarse a sí misma, de igual forma que lo hace cuando llama a otra función.

17. En el editor de Python escriba el siguiente código:

```
def cuenta_regresiva(numero):
    numero -= 1
    if numero > 0:
        print(numero)
        cuenta_regresiva(numero)
    else:
        print("...........!")
    print("Fin de la función", numero)

cuenta_regresiva(4)
```

18. Guárdelo con el nombre de actividad9.py. Ejecute el programa con la opción Run.

EJERCICIOS PROPUESTOS

EJERCICIOS PROPUESTOS

1. Crear una excepción para la validación de una división entre cero y la validación en el ingreso permitido de soló valores numéricos.

2. Modificar la función impresión de la página 4 para que el texto a enmarcarse sea elegible por el usuario como un argumento adicional. Utilice la función len(), que determina la longitud de una cadena y deje un espacio entre el texto y el símbolo para ambos lados.

3. Crear una función que permita obtener el valor de x en función de dos parámetros a y b. x=a2+2ab +b2.

4. Crear una función que permita obtener el valor de raíz cuadrada (z) de x2+y2.

5. Crear una función que calcule el IMC de acuerdo al peso y la altura. IMC = peso/ altura2.

6. Determine a través de una función si un numero entero ingresado es par o impar.

7. Calcule la suma de los n números utilizando una función de argumentos arbitrarios.

8. Elaborar una función que reciba n números enteros y nos retorne su promedio.

EJERCICIOS PROPUESTOS

9. Realiza una función llamada relacion(a, b) que a partir de dos números cumpla lo siguiente:
   - Si el primer número es mayor que el segundo, debe devolver 1.
   - Si el primer número es menor que el segundo, debe devolver -1.
   - Si ambos números son iguales, debe devolver un 0.
10. Desarrollar una función que nos retorne el perímetro de un rectángulo pasando como parámetro la base y la altura del rectángulo.
11. Crear una función que permita el ingreso del radio de un círculo y calcule el área.
12. Programar una función que valide un número celular ingresado (debe ser numérico y con 9 dígitos).
13. Escribir un programa que permita al usuario obtener un identificador para cada uno de los socios de un club (código). Para eso ingresará nombre completo y número de DNI de cada socio, indicando que finalizará el procesamiento mediante el ingreso de un nombre vacío.

EJERCICIOS PROPUESTOS

El formato del nombre de los socios será: nombre apellido la primera letra en mayúscula. Se debe validar que el número de DNI tenga 8 dígitos. En caso contrario, el programa debe dejar al usuario en un bucle hasta que ingrese un DNI correcto. Se recomienda hacer uso de un diccionario.

Ejemplo:

Código: 1

Nombre: María Linares

DNI: 25834910

14. Crear una función que realice la suma de cinco números utilizando solo *args. Debe imprimir el resultado en la consola. La suma no se realizará directamente sobre el print().
15. Definir por asignación una lista de enteros en el bloque principal del programa. Elaborar tres funciones, la primera recibe la lista y retorna la suma de todos sus elementos, la segunda recibe la lista retornando el mayor valor y la última recibe la lista retornando el menor valor.

EJERCICIOS PROPUESTOS

16. Programar una función que permita determinar si un número es primo o no.
17. Desarrollar la función recursiva Fibonacci.
18. Crear una función recursiva para el cálculo del factorial de un determinado número.
19. Haciendo uso de argumentos arbitrarios cree una función que concatene todos los argumentos arbitrarios con el separador que se le envíe como parámetro.
20. Haciendo uso del desempaquetado de parámetros hallar el mayor y el menor de 5 números.
21. Realizar una función que permita la carga de n alumnos. Por cada alumno se deberá preguntar el nombre completo (nombres y apellidos) y permita el ingreso de 3 notas. Las notas deben estar comprendidas entre 0 y 20. Devolver el listado de alumnos.
22. Desarrollar un programa que permita adivinar el número en el rango de 0 a 99, para lo cual debe crear la función adivinar, la consola debe mostrar mensajes: demasiado pequeño, muy grande o ha ganado cuando adivina el número.
23. Al ejercicio anterior añada los parámetros mínimo y máximo para controlar el número a adivinar. Utilice las funciones que considere.

EJERCICIOS PROPUESTOS

16. Programar una función que permita determinar si un número es primo o no.
17. Desarrollar la función recursiva Fibonacci.
18. Crear una función recursiva para el cálculo del factorial de un determinado número.
19. Haciendo uso de argumentos arbitrarios cree una función que concatene todos los argumentos arbitrarios con el separador que se le envíe como parámetro.
20. Haciendo uso del desempaquetado de parámetros hallar el mayor y el menor de 5 números.
21. Realizar una función que permita la carga de n alumnos. Por cada alumno se deberá preguntar el nombre completo (nombres y apellidos) y permita el ingreso de 3 notas. Las notas deben estar comprendidas entre 0 y 20. Devolver el listado de alumnos.
22. Desarrollar un programa que permita adivinar el número en el rango de 0 a 99, para lo cual debe crear la función adivinar, la consola debe mostrar mensajes: demasiado pequeño, muy grande o ha ganado cuando adivina el número.
23. Al ejercicio anterior añada los parámetros mínimo y máximo para controlar el número a adivinar. Utilice las funciones que considere.

EJERCICIOS PROPUESTOS

24. Programe una función que genere la tabla de multiplicar del 1 al 10 de un determinado número.
25. Desarrollar una función que permita sumar todos los dígitos de un número.

CUESTIONARIO

CUESTIONARIO

1. Desarrollar los ejercicios del codingbat - http://codingbat.com/python
2. ¿Qué es una función en Python?
3. ¿Cuál es la sintaxis básica para definir una función en Python?
4. ¿Qué es el valor de retorno de una función?
5. ¿Cómo se llama el valor que se pasa a una función cuando se la llama?
6. ¿Cuál es la diferencia entre parámetros y argumentos de una función?
7. ¿Qué es el ámbito (scope) de una variable en Python?
8. ¿Qué es una función lambda y cuál es su uso común?
9. ¿Cómo se pueden documentar funciones en Python?
10. ¿Qué es la recursión en programación y cómo se utiliza en funciones?

GRACIAS!