

2K DESARROLLO DE APLICACIONES INFORMÁTICAS

14/11/2025

UT1: "GIT, pseudocódigo, diagramas de flujo"

Modelo B

Tiempo: 1h40

Nombre y apellidos:

Calificación:

/10

1. Test (6 puntos)	3. Ejercicio 1 (2 puntos)	4. Ejercicio 2 (2 puntos)

Utiliza bolígrafo azul o negro.

TEST

Respuesta correcta = 0,30 puntos. Respuesta incorrecta = -0,10 puntos.

Todos los resultados que no estén en esta tabla no se considerarán una respuesta corregible.

Cualquier carácter que no pueda ser reconocible por el profesor se considerará cómo error.

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	

- 1.1. La zona temporal de intercambio o **staging-area** es:
- a- Un directorio que contiene una copia de una versión concreta del proyecto en la que se está trabajando
 - b- Es una zona donde se guardan los cambios temporalmente
 - c- Es donde finalmente se guardan los cambios confirmados
 - d- Es el repositorio remoto
- 1.2. La instrucción **GIT ADD**.
- a- Añade los cambios en el fichero <fichero> del directorio de trabajo a la zona de intercambio temporal
 - b- Añade los cambios en todos los ficheros de la carpeta <carpeta> del directorio de trabajo a la zona de intercambio temporal
 - c- Añade los cambios al repositorio remoto
 - d- Añade todos los cambios de todos los ficheros no guardados aún en la zona de intercambio temporal
- 1.3. ¿Qué instrucción GIT confirma los cambios de zona de intercambio y los añade al repositorio creando una nueva versión del proyecto?
- a- Git commit
 - b- Git push
 - c- Git init
 - d- Git add

1.4. La instrucción **code** . permite:

- a- Abrir un archivo desde terminal
- b- Abrir varios archivos desde terminal
- c- Abrir todos los archivos de la URL que hemos añadido en el explorador
- d- Todas son correctas

1.5. Visual Studio Code es:

- a- Un editor de código fuente
- b- Un controlador de versiones
- c- Un control de versiones en la nube
- d- Todas son correctas

1.6. Una función en Python

- a- Usa un espacio indentado
- b- No usa indentación
- c- Usa dos espacios indentados
- d- Se usa el tabulador para indentar

1.7. El siguiente programa devolverá:

```
def area_triangulo(base, altura):  
    #  
    return base * altura / 2  
    #  
  
#  
area_triangulo(2, 3)  
#
```

- a- El área de un triángulo de altura 2 y base 3
- b- El área de un triángulo de base 2 y altura 3
- c- No devolverá nada
- d- Almacenará en una variable el valor del área, pero no lo mostrará por pantalla

1.8. La instrucción assert en Python:

- a- Permite generar funciones
- b- Es una técnica de debugging
- c- Es un framework
- d- Permite comprobar si una condición es verdadera o no y dará un mensaje de error

1.9 Si quiero crear un bucle for, con un rango que vaya de 0 a 10 con incremento de 1, será de la siguiente forma:

- a- For i in range (1, 10, 1):
- b- For i in range (0, 10, 1):
- c- For i in range (0, 1, 10):
- d- For i in range (1, 0, 10):

1.10 Para definir una función:

- a- `def <nombre-funcion>:`
- b- `def (<parámetros>):`
- c- `def <nombre-funcion>(<parámetros>):`
- d- `def (<parámetros>)<nombre-funcion>:`

1.11 La instrucción **Git clone**, sirve para:

- a- Hacer una copia local del repositorio remoto
- b- Subir los cambios realizados en el repositorio local al remoto
- c- Realizar cambios en la zona de intercambio
- d- Añadir los cambios al repositorio local

1.12 Si quiero comprobar el estado de mi repositorio local, utilizaré la instrucción:

- a- Git add .
- b- Git push
- c- Git status
- d- Git init

1.13 git config – global user.name

- a- Establece el correo del usuario
- b- Establece el coloreado de linea
- c- Establece el nombre de usuario
- d- Muestra la configuración

1.14 git init

- a- Crea la carpeta oculta .git que contiene la base de datos donde se registran los cambios del repositorio
- b- Inicializa un repositorio nuevo
- c- Realiza cambios en la zona de intercambio
- d- Inicializa un nuevo repositorio desde el inicio creando varias carpetas en su interior

1.15 El código siguiente devolverá:

```
def bienvenida(nombre, apellido):  
#  
    print('¡Bienvenido a Python', nombre, apellido + '!')  
#  
    return  
#  
#  
bienvenida(apellido='Ugarte', nombre='Ramón')  
#
```

- a- ¡Bienvenido a Python <nombre, apellidos>!
- b- ¡Bienvenido a Python nombre apellidos!
- c- ¡Bienvenido a Python Ugarte Ramón!
- e- ¡Bienvenido a Python Ramón Ugarte!

1.16 Cuando trabajamos con funciones anidadas en Python debemos usar para diferenciar los distintos niveles de anidamiento:

- a- la indentación
- b- los espacios
- c- las líneas en blanco
- d- los comentarios

1.17. La estructura fundamental de la programación estructurada que permite la repetición es:

- a- la secuencia
- b- la iteración
- c- la alternativa
- d- cualquiera de las tres

1.18. ¿Cuál de las siguientes frases NO es correcta?

Las listas son:

- a- Un tipo de dato estructurado
- b- Tienen orden.
- c- Son inmutables, es decir, no pueden alterarse durante la ejecución de un programa.
- d- Pueden contener elementos de distintos tipos.

1.19. ¿Cuál de las siguientes frases NO es correcta?

Las Tuplas:

- a- Tienen orden.
- b- Pueden contener elementos de distintos tipos.
- c- Son mutables, es decir, pueden alterarse durante la ejecución de un programa.
- d- Se usan habitualmente para representar colecciones de datos de una determinada estructura semántica, como por ejemplo un vector o una matriz.

1.20. ¿Cuál de las siguientes es una operación que NO modifica una lista?

- a- Append
- b- Index
- c- Extend
- d- Insert

Ejercicio 1

Diseña un algoritmo que pida al usuario un número entero del 0 al 9 y:

- si el número no es correcto, vuelva a solicitar introducir un número,
- si el número es correcto, lo vaya sumando a los números previamente introducidos por el usuario, volviendo a solicitar introducir otro número,
- si el número es 99, salga del programa, devolviendo el valor de la suma total de los números correctos previamente introducidos.

Pseudocódigo (1 punto)

Diagrama de flujo (1 punto)

Ejercicio 2

Corrige los errores del código siguiente.

Código corregido:

```
# Crear una lista vacía para almacenar los números
numeros_ganadores = []

# Pedir los números al usuario
print("Introduce los 6 números ganadores de la lotería
primitiva.")
for i in range(6):
    while True:
        try:
            # Pedir un número y guardarlo como entero
            numero = int((f"Introduce el número {i+1}: "))
            # Añadir el número a la lista
            numeros_ganadores.append(numero)
            # Salir del bucle while si el número es válido
        except ValueError:
            print("Por favor, introduce un número válido.")

# Ordenar la lista de números
numeros_ganadores.sort()

# Mostrar los números ordenados
print("\nLos números ganadores ordenados de menor a mayor son:")
```

Código corregido: