Examen Entornos de Desarrollo

20 Febrero, 2024

**Nombre y apellidos: \_\_SAMUEL PLAZA SÁEZ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1. Se va a realizar una entrada de datos de un médico en un hospital por pantalla gráfica, se definen 3 campos de entrada y una lista para elegir la especialidad. La aplicación acepta los datos de esta manera:**

1. Idmédico: número de 4 dígitos que no empiece por 0.
2. Apellido: cadena de 20 caracteres como máximo.
3. Consulta: en blanco o número entero entre 1 y 20.
4. Especialidad: Cardiología, Pediatría, Dermatología, Neurología o Selecciona Especialidad.

Se debe rellenar la tabla de clases de equivalencia y las tablas de los casos de prueba correspondientes usando la técnica de particiones o clases de equivalencia.

**Condiciones de entrada para el supuesto, junto con las clases de equivalencia válidas y no válidas de acuerdo con las normas especificadas (1,15 puntos)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Condición de entrada** | **Tipo** | **Clases válidas** | **Clases no válidas** |
| **Idmédico** | **Rango** | 1.000 <= Idmédico <= 9.999 (1) | Idmédico < 1.000 (2)  Idmédico >9.999 (3)  Idmédico != número (4) |
| **Apellido** | **Valor** | Apellido max 20 caracteres (5) | Apellido > 20 caracteres (6)  Apellido en blanco (7) |
| **Consulta** | **Lógica**  **Rango** | En blanco (8)  1 <= Consulta <=20 (9) | Consulta <1 (10)  Consulta >20 (11)  Consulta != número entero (12) |
| **Especialidad** | **Miembro de un conjunto** | Cardiología (13)  Pediatría (14)  Dermatología (15)  Neurología (16) | Selecciona especialidad (17) |

**Casos de Prueba con Clases de Equivalencia Válidas (1,15 puntos)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Idmédico** | **Apellido** | **Consulta** | **Especialidad** | **Clases Incluidas** |
| 3.446 | Plaza |  | Cardiología | (1),(5),(8),(13) |
| 3.446 | Plaza | 13 | Cardiología | (1),(5),(9),(13) |
| 3.446 | Plaza |  | Pediatría | (1),(5),(8),(14) |
| 3.446 | Plaza | 13 | Pediatría | (1),(5),(9),(14) |
| 3.446 | Plaza |  | Dermatología | (1),(5),(8),(15) |
| 3.446 | Plaza | 13 | Dermatología | (1),(5),(9),(15) |
| 3.446 | Plaza |  | Neurología | (1),(5),(8),(16) |
| 3.446 | Plaza | 13 | Neurología | (1),(5),(9),(16) |

**Casos de Prueba con Clases de Equivalencia No Válidas (1,2 puntos)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Idmédico** | **Apellido** | **Consulta** | **Especialidad** | **Clases Incluidas** |
| 743 | Plaza | 13 | Cardiología | (2),(5),(9),(13) |
| 45.823 | Plaza | 13 | Cardiología | (3),(5),(9),(13) |
| fgnh | Plaza | 13 | Cardiología | (4),(5),(9),(13) |
| 3.446 | Plaza Plaza Plaza Plaza Plaza | 13 | Cardiología | (1),(6),(9),(13) |
| 3.446 |  | 13 | Cardiología | (1),(7),(9),(13) |
| 3.446 | Plaza | -1 | Cardiología | (1),(5),(10),(13) |
| 3.446 | Plaza | 63 | Cardiología | (1),(5),(11),(13) |
| 3.446 | Plaza | ofgnb | Cardiología | (1),(5),(12),(13) |
| 3.446 | Plaza | 13 | Selecciona especialidad | (1),(5),(9),(17) |

Partiendo de la condición de entrada Idmédico, indica qué clases de equivalencia adicionales habría que crear, y genera los casos de prueba correspondientes empleando la técnica de análisis de valores límite.

**Clases de equivalencia válidas y no válidas para la condición de entrada Idmédico empleando la técnica de análisis de valores límite (0,5 puntos)**

|  |  |  |
| --- | --- | --- |
| **Condición de entrada** | **Clases válidas** | **Clases no válidas** |
| **Idmédico** | Idmédico = 1.000 (18)  Idmédico = 9.999 (19) | Idmédico = 999 (20)  Idmédico = 10.000 (21) |

**Casos de prueba con clases de equivalencia válidas que se generan empleando la técnica de análisis de valores límite (0,5 puntos)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Idmédico** | **Apellido** | **Consulta** | **Especialidad** | **Clases Incluidas** |
| 1.000 | Plaza | 13 | Cardiología | (18),(5),(9),(13) |
| 9.999 | Plaza | 13 | Cardiología | (19),(5),(9),(13) |

**Casos de prueba con clases de equivalencia no válidas que se generan empleando la técnica de análisis de valores límite (0,5 puntos)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Idmédico** | **Apellido** | **Consulta** | **Especialidad** | **Clases Incluidas** |
| 999 | Plaza | 13 | Cardiología | (20),(5),(9),(13) |
| 10.000 | Plaza | 13 | Cardiología | (21),(5),(9),(13) |

**2. Depura el siguiente código teniendo en cuenta las indicaciones que se detallan a continuación:**

def llenar(n):

    tabla = [0] \* n

    for i in range(n):

        tabla[0] = i \* 10

    return tabla

def sumar(tabla):

    suma = 0

    n = len(tabla)

    for i in range(n-1):

        suma -= tabla[i]

    return suma

def main():

    n = 5

    tabla = [0] \* n

    tabla = llenar(n)

    suma = sumar(tabla)

    print("La suma es:", suma)

# Llamar a la función principal

if \_\_name\_\_ == "\_\_main\_\_":

    main()

1. La función llenar() recibe el parámetro entero n y devuelve una lista con n enteros. La segunda función llamada sumar() recibe una lista de enteros, suma sus elementos y devuelve la suma.
2. Coloca un punto de ruptura en la línea tabla = llenar(n). Realiza una captura de pantalla. (0,5 puntos)

Captura de pantalla de computadora

Descripción generada automáticamente

1. Con el botón correspondiente entra en la función llenar(). Realiza una captura de pantalla. (0,5 puntos)

Captura de pantalla de computadora

Descripción generada automáticamente

1. Muestra cuando la variable i=2 y la variable n=5. Muestra el contenido de tabla. Realiza una captura de pantalla. (0,5 puntos)

Captura de pantalla de computadora

Descripción generada automáticamente

1. Muestra la longitud de tabla. Realiza una captura de pantalla. (0,5 puntos)

Captura de pantalla de computadora

Descripción generada automáticamente

1. Continúa la depuración hasta corregir todos los errores del código. Realiza una captura de pantalla del código corregido. (2 puntos)

Texto

Descripción generada automáticamente

**3. Depura el siguiente código para que muestre los elementos de la lista en orden ascendente:**

import random

def ordenar\_lista(numeros):

    n = len(numeros)

    lista\_ordenada = numeros

    for i in range(n):

        for j in range(0, n-i-1):

            if lista\_ordenada[j+1] > lista\_ordenada[j]:

                lista\_ordenada[j], lista\_ordenada[j+1] = lista\_ordenada[j+1], lista\_ordenada[j]

    return numeros

numeros = []

for i in range(5):

    numeros.append(random.randint(1,30))

print("La lista original es: ",(numeros))

print("La lista ordenada es: ",ordenar\_lista(numeros))

1. Realiza una captura de pantalla donde se muestren los elementos de la lista original. (0,2 puntos)

Captura de pantalla de computadora

Descripción generada automáticamente

1. Realiza una captura de pantalla con el código corregido y depurado. (0,8 puntos)

Captura de pantalla de computadora

Descripción generada automáticamente

**Nota:**

**Se evalúa el RA3 del módulo.**