Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ciencias de la Computación
CC1002-10 – Introducción a la Programación

Pauta Laboratorio #3

Recursividad II

Profesor: Patricio Inostroza
Auxiliares: Fabián Cruz, Fernando Palma, Juan Valdivia, Nicolas Wiertz

P1. Fractales

Para resolver este problema, bastaba con modificar el código entregado para el fractal de Koch, agregando las instrucciones necesarias para dibujar el cuadrado que se proyecta hacia afuera en cada segmento dividido en tres partes. El código completo es el siguiente:

```
#fractalCuadrado.py

import turtle

#lado: int int (int) -> None
#dibuja lado de fractal cuadrado
#de nivel n y largo L(minimo Lmin)
def lado(n, L, Lmin=6):
    assert type(n)==int and n>=1
    assert (type(L)==int or type(L)==float) and L>0
    if n==1 or L<Lmin:
        turtle.forward(L)
    else:
        # Dividir el lado en tres partes
        # Primera parte: 1/3 del lado original
        lado(n-1, L/3)

        # Dibujar el cuadrado que se proyecta hacia afuera
        turtle.left(90)   # Girar 90° a la izquierda
        lado(n-1, L/3)    # Lado vertical del cuadrado
        turtle.right(90)  # Volver a la dirección original
        lado(n-1, L/3)    # Lado horizontal superior del cuadrado
        turtle.right(90)  # Girar 90° a la derecha
```

```
        lado(n-1, L/3)    # Lado vertical derecho del cuadrado
        turtle.left(90)   # Volver a la dirección original

        # Tercera parte: último 1/3 del lado original
        lado(n-1, L/3)

#fractal: int int -> None
#dibuja fractal cuadrado de nivel n y lado L
def fractal(n, L):
    assert type(n)==int and n>=1
    assert (type(L)==int or type(L)==float) and L>0

    lado(n, L)       # Lado 1
    turtle.right(90) # Giro de 90° ↱
    lado(n, L)       # Lado 2
    turtle.right(90) # Giro de 90° ↱
    lado(n, L)       # Lado 3
    turtle.right(90) # Giro de 90° ↱
    lado(n, L)       # Lado 4
    # No necesita girar 90° final porque ya completó el cuadrado

fractal(3, 200) # Ejecución
```

P2. “Blackjack”

La idea principal de este problema es implementar funciones recursivas para manejar el turno del jugador y del crupier, así como un menú principal que permita al usuario jugar múltiples veces o salir del programa.

Como el blackjack es un juego en el que se repiten turnos hasta que el jugador decida plantarse o se pase de 21, la recursión es una herramienta adecuada para este caso.

A continuación se presenta el código completo con las funciones propuestas:

1ra Función

```python
def sacar_carta():
    # Saca una carta aleatoria (número del 1 al 11)
    return random.randint(1, 11)
```

2da Función

```python
def jugar_turno(puntuacion_actual):
    # Turno del jugador con recursión
    print(f"\nTu puntuación actual: {puntuacion_actual}")

    if puntuacion_actual > 21:
        print("¡Te pasaste de 21! Perdiste.")
        return puntuacion_actual
    elif puntuacion_actual == 21:
        print("¡Perfecto! Llegaste a 21.")
        return puntuacion_actual

    decision = input("¿Quieres sacar otra carta? (s/n): ").lower().strip()

    if decision == 's':
        nueva_carta = sacar_carta()
        print(f"Sacaste: {nueva_carta}")
        nueva_puntuacion = puntuacion_actual + nueva_carta
        return jugar_turno(nueva_puntuacion)  # Recursión
    else:
        print(f"Te quedas con: {puntuacion_actual}")
        return puntuacion_actual
```

3ra Función

```python
def jugar_crupier(puntuacion_crupier):
    # El crupier juega usando recursión
    print(f"Crupier tiene: {puntuacion_crupier}")

    if puntuacion_crupier >= 17:
        return puntuacion_crupier
    else:
        nueva_carta = sacar_carta()
        print(f"Crupier saca: {nueva_carta}")
        return jugar_crupier(puntuacion_crupier + nueva_carta)  # Recursión
```

4ta Función

```python
def determinar_ganador(puntuacion_jugador, puntuacion_crupier):
    # Determina quién ganó el juego
    print("\n" + "==================================================")
    print("RESULTADO FINAL")
    print("==================================================")
    print(f"Tu puntuación: {puntuacion_jugador}")
    print(f"Crupier: {puntuacion_crupier}")
    print("--------------------------------------------------")

    if puntuacion_jugador > 21:
        print("PERDISTE - Te pasaste de 21")
    elif puntuacion_crupier > 21:
        print("¡GANASTE! - El crupier se pasó")
    elif puntuacion_jugador > puntuacion_crupier:
        print("¡GANASTE! - Tienes mayor puntuación")
    elif puntuacion_crupier > puntuacion_jugador:
        print("PERDISTE - El crupier tiene mayor puntuación")
    else:
        print("¡EMPATE!")
    print("==================================================")
```

5ta Función

```
def jugar_blackjack():
    # Función principal del juego
    print("\n" + "==============================================")
    print(" BLACKJACK ")
    print("==============================================")
    print("Objetivo: Llegar lo más cerca posible a 21")
    print("Las cartas van del 1 al 11")
    print("==============================================")

    # Primera carta del jugador
    primera_carta = sacar_carta()
    print(f"\nTu primera carta: {primera_carta}")

    # Jugar turno del jugador
    puntuacion_jugador = jugar_turno(primera_carta)

    # Solo si el jugador no se pasó, juega el crupier
    if puntuacion_jugador <= 21:
        print("\nTurno del crupier:")
        primera_carta_crupier = sacar_carta()
        puntuacion_crupier = jugar_crupier(primera_carta_crupier)
        determinar_ganador(puntuacion_jugador, puntuacion_crupier)
    else:
        print("\n" + "==============================================")
        print("PERDISTE - Te pasaste de 21")
        print("==============================================")

    return puntuacion_jugador
```

6ta Función

```
def menu_principal():
    # Muestra el menú principal
    print("\n MENÚ PRINCIPAL ")
    print("1. Jugar Blackjack")
    print("2. Salir")
    print("----------------------------------------------")
```

```
    opcion = input("Elige una opción (1-2): ").strip()

    if opcion == "1":
        jugar_blackjack()
        input("\nPresiona Enter para continuar...")
        menu_principal()  # Recursión: volver al menú
    elif opcion == "2":
        print("\n ¡Gracias por jugar!")
        return  # Salir del programa
    else:
        print("\n Opción no válida. Elige 1 o 2.")
        menu_principal()  # Recursión: volver al menú


menu_principal() # Inicia el programa
```

Nota:

- La función `strip()` elimina espacios en blanco al inicio y final de una cadena. Sirve para limpiar la entrada del usuario.
- La función `lower()` convierte una cadena a minúsculas.
- Los f-strings permiten incrustar expresiones dentro de cadenas usando llaves {}. Ej: `f"Valor: {x}"`.
- `\n` representa un salto de línea en una cadena. Permite imprimir en una nueva línea.

Nota 2: Faltan las recetas de diseño para cada función. Aunque no estén aquí, nunca las olviden en sus evaluaciones :D