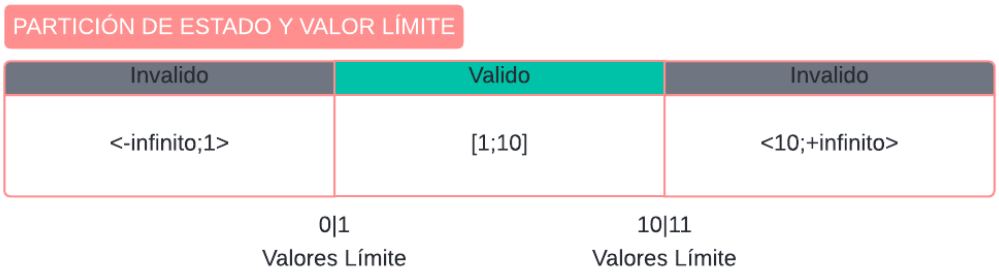
PARCIAL 2 DE PRUEBAS DE SOFTWARE

Nombre: Renzo Andre Dejo Verdi.

Partición de equivalencias y valores límites:

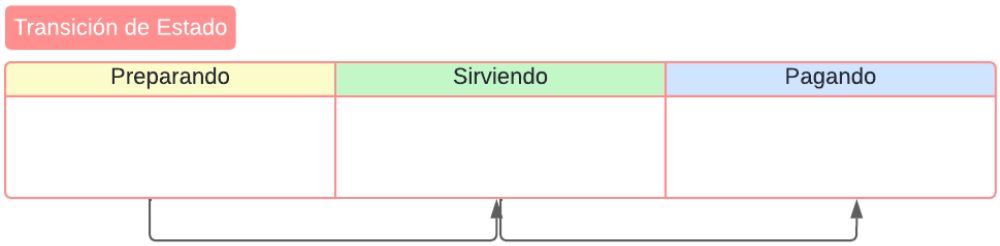



```python
def reserve_table(self, num_people):
    # Partición de equivalencias y valores límite
    if not (1 <= num_people <= 10):
        return "Error: Invalid number of people. Must be between 1 and 10."
    for i in range(len(self.tables)):
        if self.tables[i] == 0:
            self.tables[i] = num_people
            return f"Table {i + 1} reserved for {num_people} people."
    return "Error: No tables available."

def create_order(self, table_number, order_details):
    if table_number < 1 or table_number > 10 or self.tables[table_number - 1] == 0:
        return "Error: Invalid or unreserved table."
    order = {
        'table': table_number,
        'details': order_details,
        'state': 'CREATED'
    }
    self.orders[self.order_id] = order
    self.order_id += 1
    return f"Order {self.order_id - 1} created."
```

En esta parte del código se muestra que coloco como valores para la partición de equivalencias, que en este caso he puesto 3 particiones y 4 valores límite como se muestra en el diagrama , y abajo adjunto prueba de como se aplica con el valor 4 para la partición de equivalencia porque está dentro del rango, y 11 como prueba de valor límite, que mostraría error.

```
68    print(restaurant_system.reserve_table(4))
69    print(restaurant_system.reserve_table(11))
70
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   COMMENTS

 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
PS D:\CURSOS 2024-I\Pruebas de Software\Parcial2> python restaurante.py
Table 1 reserved for 4 people.
Error: Invalid number of people. Must be between 1 and 10.
```

Transición de Estado:




```python
def update_order_state(self, order_id, new_state):
    # Transición de Estado
    valid_transitions = {
        'CREATED': ['PREPARING', 'CANCELLED'],
        'PREPARING': ['SERVED', 'CANCELLED'],
        'SERVED': ['PAID'],
        'CANCELLED': [],
        'PAID': []
    }
    if order_id not in self.orders:
        return "Error: Invalid order ID."
    current_state = self.orders[order_id]['state']
    if new_state in valid_transitions[current_state]:
        self.orders[order_id]['state'] = new_state
        return f"Order {order_id} state updated to {new_state}."
    else:
        return f"Error: Invalid state transition from {current_state} to {new_state}."
```

En esta parte del código muestro los estados que pueden incluir una orden y la secuencia que debe tomar.

```python
# Actualizar estado de la orden
print(restaurant_system.update_order_state(1, 'PREPARING'))
print(restaurant_system.update_order_state(1, 'PAID'))
```

```
Order 1 state updated to PREPARING.
Error: Invalid state transition from PREPARING to PAID.
```

Aquí se muestra el estado de la orden que ingreso, y su respectiva respuesta, en la 2da es correcto que muestre error, ya que no puede pasar del estado “PREPARING” al estado “Paid”, antes tiene que pasar por el estado “SERVED”

Tablas de Decisiones

Tabla de Desiciones

WELCOME10	SUMMER20	VIP30
precio=precio*0.10	precio=precio*0.20	precio=precio*0.30

```python
def apply_discount(self, order_id, discount_code):
    # Tablas de decisiones
    discount_rates = {
        'WELCOME10': 0.10,
        'SUMMER20': 0.20,
        'VIP30': 0.30
    }
    if order_id not in self.orders:
        return "Error: Invalid order ID."
    if discount_code not in discount_rates:
        return "Error: Invalid discount code."
    order = self.orders[order_id]
    original_price = sum(item['price'] for item in order['details'])
    discount_rate = discount_rates[discount_code]
    discounted_price = original_price * (1 - discount_rate)
    return f"Original price: ${original_price:.2f}, Discounted price: ${discounted_price:.2f}"
```

En esta parte del código muestro los tipos de descuento que se hará acorde al código ingresado, según el código ingresado se realizará un respectivo descuento, así mismo, mostraré los resultados acordes a las pruebas realizadas.

```python
print(restaurant_system.apply_discount(1, 'WELCOME10'))
print(restaurant_system.apply_discount(1, 'INVALIDCODE'))
```

```
Original price: $12.00, Discounted price: $10.80
Error: Invalid discount code.
```

Con el código WELCOME10 se realiza un descuento del 10%, cosa que es correcto el resultado, y abajo se puso un código inexistente, por eso está bien que muestre el error.