Informe Técnico: Sistema Organizador de Tareas en SWI-Prolog

Objetivo del Proyecto

El sistema implementa un organizador de tareas que permite gestionar actividades diarias con diferentes niveles de prioridad y fechas límite. El programa está diseñado para ayudar en la organización personal, priorizando automáticamente las tareas según su urgencia e importancia.

Mecanismos de Prolog Utilizados

1. Base de Conocimiento Dinámica

prolog
Copy

```
:- dynamic tarea/5.
```

Este mecanismo permite:

- Modificar la base de datos en tiempo de ejecución
- Agregar y eliminar tareas dinámicamente
- Mantener el estado del programa

2. Predicados de Control

prolog
Copy

```
% Ejemplo de control con if-then-else
(Importancia >= 1, Importancia =< 5 ->
   assert(tarea(ID, Descripcion, FechaLimite, pendiente, Importancia))
;
   writeln('Error: El nivel de importancia debe estar entre 1 y 5'))
```

3. Manipulación de Listas

prolog
Copy

```
findall(tarea(ID, Desc, Fecha, Estado, Imp),
        tarea(ID, Desc, Fecha, Estado, Imp),
    Tareas)
```

Reglas de Inferencia

1. Ordenamiento de Tareas

prolog
Copy

```
comparar_tareas(Orden, tarea(ID1,_,Fecha1,_,Imp1), tarea(ID2,_,Fecha2,_,Imp2))
```

Establece la lógica para:

- Comparar fechas límite
- Evaluar niveles de importancia
- Determinar el orden final de las tareas

2. Filtrado de Tareas

prolog
Copy

```
es_pendiente(tarea(_,_,_,pendiente,_)).
```

Define las condiciones para:

- Identificar tareas pendientes
- Filtrar elementos según su estado

Ventajas del Enfoque

1. **Programación Declarativa**
   - Código más legible y mantenible
   - Separación clara entre lógica y datos
   - Facilidad para modificar reglas de negocio
2. **Gestión de Estado**
   - Modificación dinámica de la base de conocimiento
   - Persistencia temporal de datos
   - Consultas flexibles
3. **Manejo de Datos**
   - Estructuras de datos uniformes
   - Fácil extensibilidad
   - Búsquedas eficientes

Explicación Detallada de Funciones

1. Gestión de Entrada/Salida

leer_texto/1

prolog
Copy

```
leer_texto(Texto) :-
    read_line_to_string(user_input, Texto).
```

Uso: Lee texto con espacios del usuario **Función**: Maneja la entrada de descripciones de tareas

limpiar_buffer/0

prolog
Copy

```
limpiar_buffer :-
    read_line_to_string(user_input, _).
```

Uso: Limpia el buffer de entrada **Función**: Previene errores de lectura

2. Gestión de Tareas

agregar_tarea/4

prolog
Copy

```
agregar_tarea(ID, Descripcion, FechaLimite, Importancia)
```

Uso: Agrega una nueva tarea al sistema **Parámetros**:

- ID: Identificador único
- Descripcion: Texto descriptivo
- FechaLimite: Fecha en formato AAAA-MM-DD
- Importancia: Valor de 1 a 5

completar_tarea/1

prolog
Copy

```
completar_tarea(ID)
```

Uso: Marca una tarea como completada **Función**: Actualiza el estado de pendiente a completada

eliminar_tarea/1

prolog
Copy

```
eliminar_tarea(ID)
```

Uso: Elimina una tarea del sistema **Función**: Remueve permanentemente la tarea de la base de datos

3. Sistema de Ordenamiento

ordenar_tareas/1

prolog
Copy

```
ordenar_tareas(TareasOrdenadas)
```

Uso: Ordena las tareas por fecha y prioridad **Proceso**:

1. Recolecta todas las tareas
2. Filtra las pendientes
3. Aplica el ordenamiento personalizado

fecha_a_termino/2

prolog
Copy

```
fecha_a_termino(Fecha, Termino)
```

Uso: Convierte fechas a términos comparables **Función**: Facilita la comparación de fechas

4. Visualización

listar_tareas/0

prolog
Copy

```
listar_tareas
```

Uso: Muestra todas las tareas en el sistema **Formato**: ID | Descripción | Fecha | Estado | Importancia

listar_pendientes/0

prolog
Copy

```
listar_pendientes
```

Uso: Muestra solo tareas pendientes ordenadas **Ordenamiento**: Por fecha límite y prioridad

5. Búsqueda

buscar_tarea/1

prolog
Copy

```
buscar_tarea(Palabra)
```

Uso: Busca tareas por palabra clave **Función**: Encuentra coincidencias en descripciones

6. Control de Programa

menu/0

prolog
Copy

```
menu
```

Uso: Muestra y gestiona el menú principal **Opciones**:

1. Agregar tarea
2. Marcar como completada
3. Eliminar tarea
4. Listar todas
5. Listar pendientes
6. Buscar
7. Salir

inicio/0

prolog
Copy

```
inicio
```

Uso: Punto de entrada al programa **Función**: Inicia el ciclo del menú principal

Ejemplos de Uso

prolog
Copy

```
% Iniciar el programa
?- inicio.

% Agregar una tarea
ID de la tarea (número): 1
Descripción: Completar informe mensual
Fecha límite (AAAA-MM-DD): 2024-10-25
Importancia (1-5): 4

% Listar tareas pendientes
?- listar_pendientes.

% Completar una tarea
?- completar_tarea(1).
```

Consideraciones Técnicas

1. **Validación de Datos**
    - Control de niveles de importancia (1-5)
    - Verificación de formato de fechas
    - Unicidad de IDs
2. **Manejo de Errores**
    - Control de entradas inválidas
    - Gestión de casos límite
    - Mensajes de error informativos
3. **Persistencia**
    - Los datos se mantienen en memoria durante la sesión
    - Se pierden al cerrar Prolog
    - Posibilidad de extensión para almacenamiento permanente