Ejercicios de Punteros con Pascal

Información del Proyecto

Descripción	Detalles
Profesores	Sergio Cavero y Salvador Sanchez
Asignatura	Estructuras de Datos
Universidad	Universidad Rey Juan Carlos

Ejercicio 1 (Nivel básico)

Sigue la siguiente secuencia de ejecución escribiendo código Pascal y a la vez dibuja en un papel lo que ocurre tras ejecutar cada instrucción [Nivel básico]:

1. Declara variable entera (x)
2. Declara una variable de tipo puntero a entero (p_ent)
3. Da el valor 100 a x
4. Crea un entero dinámicamente con p_ent y dale el valor que tiene actualmente x
5. Imprime por pantalla el valor contenido en el entero al que apunta p_ent
6. Crea con new un nuevo entero dinámicamente
7. Imprime por pantalla el valor contenido en el entero al que apunta p_ent
8. Pon el puntero p_ent a apuntar a x
9. Súmale 100 al entero apuntado por el puntero p_ent
10. Imprime por pantalla el valor de x y también del entero al que apunta p_ent
11. Pon el puntero p_ent a NIL
12. Libera la memoria asociada al nuevo entero
13. Suma 100 a x pero sin usar x en la operación de suma (solo usando p_ent)
14. Muestra por pantalla que son iguales
15. Libera toda la memoria asociada a p_ent y termina

Ejercicio 2 (Nivel básico)

Sigue la siguiente secuencia de ejecución escribiendo código Pascal y a la vez dibuja en un papel lo que ocurre tras ejecutar cada instrucción [Nivel básico]:

1. Crea un array de 3 enteros V e inicialízalo con números aleatorios.

2. Muestra sus valores por pantalla
3. Declara un puntero a entero p y ponlo a apuntar a la primera posición del array
4. Pon el valor 100 en el entero apuntado por p
5. Muestra los valores del array "V" por pantalla
6. Recorre con p todas las posiciones del array para ponerlas todas a cero
   (Nota: puedes sumar 1 a p para pasar a la siguiente)

Ejercicio 3 (Nivel medio)

Sigue la siguiente secuencia de ejecución escribiendo código Pascal y a la vez dibuja en un papel lo que ocurre tras ejecutar cada instrucción:

1. Declara un tipo de registro llamado "nodo" con dos campos: un entero y un puntero a "nodo".
2. Declara una variable de tipo puntero a nodo
3. Crea un nodo en memoria dinámica (heap)
4. Dale valor 100 al campo entero de este nodo
5. Pon a nil el campo puntero del nodo
6. Pon ahora el puntero del nodo a apuntar al propio nodo
7. En este punto hay dos instrucciones alternativas para liberar la memoria dinámica creada: identifícalas y escríbelas. Dibuja cómo quedarían las cosas tras cada una de ellas.
8. Termina

Ejercicio 4 (Nivel medio)

Sigue la siguiente secuencia de ejecución escribiendo código Pascal y a la vez dibuja en un papel lo que ocurre tras ejecutar cada instrucción:

1. Declarar un array de MAX punteros a enteros
2. Crea un entero en cada una de las posiciones inicializado a un número aleatorio
3. Implementa un procedimiento que muestre por pantalla el contenido de cada posición
4. Implementa un procedimiento que ponga a cero todos los números
5. Implementa un subprograma que ponga todos los punteros que no lo estén, a NIL
6. Implementa una función que calcule el máximo número del array (debes comprobar antes, pues puede que alguna posición no tenga un número y esté a NIL)
7. Implementa una función que compute la suma de todos los enteros (de nuevo asegúrate de no intentar sumar los que estén a NIL)

Ejercicio 5 (Nivel medio)

1. Declara un tipo coordenada_3D para puntos con coordenadas x,y,z
2. Declara un array de MAX punteros a coordenadas_3D
3. Crea una coordenada_3D en cada una de las posiciones inicializado todas las coordenadas con números aleatorios
4. Implementa un subprograma que libere la memoria asignada a una posición
5. Implementa un procedimiento que sume todas las x
6. (Opcional) Implementa un subprograma que permita añadir una coordenada en la primera posición libre del array
7. Implementa un subprograma que calcule la máxima coordenada y (debes comprobar antes, pues puede que alguna posición no tenga coordenada y esté a NIL)

Ejercicio 6 (Nivel Avanzado)

Simulador de "La Isla de las Tentaciones" en Pascal

En este ejercicio avanzado, vamos a crear un simulador del conocido programa de televisión "La Isla de las Tentaciones" utilizando Pascal. El objetivo es practicar el uso de punteros, registros, y la gestión dinámica de memoria mientras repasamos conceptos básicos de programación.

El programa simulará un reality show donde participan parejas y tentadores. Las parejas pondrán a prueba su relación mientras los tentadores intentan seducirlos. El estado de las relaciones irá cambiando a medida que se desarrolle el juego.

1. Definición de Tipos y Estructuras
   Primero, necesitarás crear un registro `Participante` que contendrá la información básica de cada persona en la isla:

- Un campo `nombre` de tipo String para almacenar el nombre del participante
- Un campo `edad` de tipo Integer para la edad
- Un campo `rol` que será un tipo enumerado con dos posibles valores: tentador o pareja
- Un campo `otroParticipante` que será un puntero a otro participante (representará su pareja actual)

2. Gestión de Memoria
   Deberás implementar:

- Una constante `NUM_PAREJAS` que determinará el número de parejas participantes
- Una constante `NUM_PARTICIPANTES` calculada como `2 * NUM_PAREJAS + NUM_PAREJAS div 2`
- Un array de punteros a Participante con tamaño `NUM_PARTICIPANTES`

3. Implementa un procedimiento para **crear un participante:**:

- Recibirá como parámetros: nombre, edad y rol
- Inicializará un nuevo participante con estos datos
- El puntero a otro participante se iniciará como nil

4. Implementa un procedimiento para **establecer la pareja de un participante**:

- Recibirá dos participantes como parámetros
- Establecerá la relación entre ambos participantes mediante punteros
- Cada participante apuntará al otro como su pareja

6. Crea un procedimiento para **inicializar la isla**:

- Por cada dos parejas debe haber un tentador
- Las parejas deben inicializarse juntas (cada miembro apuntando al otro)
- Los tentadores comienzan sin estar emparejados (puntero a nil)
- Los participantes se almacenarán en el array de manera ordenada, ¿cómo podrías hacerlo para no tener que determinar el rol de cada persona del array?

5. Implementa un procedimiento que **muestre el estado actual de la isla**, incluyendo para cada participante:

- Nombre y edad
- Su rol (pareja o tentador)
- Para participantes en pareja: el nombre de su pareja actual
- Para participantes que eran pareja pero ya no tienen pareja: mostrar "Infiel"
- Para tentadores que han conseguido emparejar: mostrar "Tentó a: [nombre]"

6. Implementa un procedimiento para **simular una ronda de tentación**:

- Selección aleatoria de una pareja para ser tentada
- Selección aleatoria de un tentador
- Decisión aleatoria de qué miembro de la pareja será tentado (50% de probabilidad)
- Probabilidad del 25% de que ocurra una infidelidad
- Actualización del estado:
    - Si hay infidelidad: la pareja original se rompe y se forma una nueva con el tentador
    - Si no hay infidelidad: solo se muestra el intento fallido

7. Finalmente, implementa el programa principal que:

- Muestre el estado actual de la isla
- Pregunte al usuario si desea continuar con otra ronda
- Ejecute rondas de tentación hasta que el usuario decida terminar
- Muestre el estado final de todas las relaciones

Consideraciones Adicionales

- Utiliza gestión dinámica de memoria (New/Dispose) apropiadamente
- Maneja los punteros con cuidado para evitar referencias inválidas
- Asegúrate de que la actualización de las relaciones mantiene la consistencia de los datos
- Incluye mensajes informativos que hagan el seguimiento del juego más interesante

Retos Opcionales

- Añade estadísticas de infidelidad por participante
- Implementa un sistema de afinidad entre participantes
- Añade eventos especiales durante las rondas de tentación
- Permite que el usuario seleccione manualmente las parejas a tentar