

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help

WHAT IS A DANGLING POINTER?

- ✓ A pointer pointing to a freed memory location or the location whose content has been deleted is called a **dangling pointer**.
- ✓ Dangling pointers arise during object destruction when an object that has an incoming reference is deleted or deallocated, without modifying the value of the pointer, so that the pointer still points to the memory location of the deallocated memory.

THE C PROGRAMMING LANGUAGE

02:16 24:02

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help

CAUSES OF DANGLING POINTER

- ✓ Deallocation of memory
- ✓ Returning local variables in function calls
- ✓ Variable going out of scope

THE C PROGRAMMING LANGUAGE

03:17 24:02

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

① DEALLOCATION OF MEMORY

- When memory is deallocated, the pointer keeps pointing to the freed space.
- Example:

```
char *str = "Hi Harry!";
int a;
int *ptr = (int *) malloc(sizeof(int));
free(ptr); // ptr now becomes a dangling pointer
ptr = NULL; // ptr no longer dangling
```

Handwritten notes: ptr (happy face) points to a memory block, ptr (sad face) points to a freed memory block (marked with an 'x').

05:40 24:02

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

② RETURNING LOCAL VARIABLES IN FUNCTION CALLS

```
#include <stdio.h>
int *myFunc() {
    // a is local variable and goes out of scope on function return over.
    int a = 34;
    return &a;
}

int main() {
    int *ptr = myFunc(); // ptr points to invalid location
    printf("%d", *ptr);
    return 0;
}
```

Handwritten notes: $int^* ptr = \text{Address}$, $myFunc()$ returns \sim , $main$ receives \sim . A red arrow points to the `myFunc()` call in `main` with the text "Dangling pointer".

09:07 24:02

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help

VARIABLE GOING OUT OF SCOPE

```
#include<stdio.h>

int main() {
    int *ptr;
    {
        int i = 0;
        ptr = &i; // ptr points to invalid location
    }
    // ptr is now a dangling pointer
    printf("%d", *ptr);
    return 0;
}
```

Handwritten notes: "Scope starts" with an arrow pointing to the opening brace of the inner block. "Scope ends" with an arrow pointing to the closing brace of the inner block. "ptr" with a sad face and an arrow pointing to the pointer variable declaration. A red circle is drawn around the closing brace of the inner block.

11:41 24:02

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help

IS DANGLING POINTER A GOOD THING TO HAVE?

Handwritten notes: "ptr" with a sad face and an arrow pointing to a circle labeled "freed".

- Dangling pointers can introduce nasty bugs in our C programs.
- Dangling pointer bugs frequently become security holes at times.
- We can avoid issues caused by dangling pointers by initializing pointer to NULL.
- After de-allocating memory, pointer will be no longer dangling.
- Assigning NULL value means pointer is a null pointer now.

Handwritten note: ptr = NULL;

THE C PROGRAMMING LANGUAGE

13:34 24:02

Dangling Pointer In C Language: C Tutorial In Hindi #54.mp4 - VLC media player

Media Playback Audio Video Subtitle Tools View Help