**Issues in the original code:**

1. **Uninitialized Pointer (ptrName):** The pointer ptrName is uninitialized, leading to undefined behavior when passed to printf.
2. **Typo (ptrSallary):** The variable ptrSallary is a typo and should be ptrSalary.
3. **Accessing Freed Memory:** After calling free(ptrSalary), the pointer is still used, which leads to undefined behavior.
4. **Incorrect Memory Allocation for alphabet:** Allocating 24 bytes, but accessing 26 elements, leads to buffer overflow.
5. **Incorrect Pointer Arithmetic:** Subtracting 26 from alphabet and accessing revAlphabet will result in accessing invalid memory.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main() {

    char \*ptrName;

    char fullName[] = "Md. Ishmam Uddin";

    ptrName = fullName;

    printf("My name is %s\n", ptrName);

    int \*ptrSalary = (int\*) malloc(sizeof(int));

    if (ptrSalary == NULL) {

        printf("Memory allocation failed\n");

        return 1;

    }

    \*ptrSalary = 1000;

    printf("My salary is %d\n", \*ptrSalary);

    free(ptrSalary);

    ptrSalary = NULL;

    ptrSalary=(int\*)malloc(sizeof(int));

    \*ptrSalary = 2000;

    printf("My salary is %d\n", \*ptrSalary);

    char\* alphabet = (char\*) malloc(26 \* sizeof(char));

    if (alphabet == NULL) {

        printf("Memory allocation failed\n");

        return 1;

    }

    for (int i = 0; i < 26; i++) {

        alphabet[i] = 'A' + i;

    }

    printf("Alphabet: %s\n", alphabet);

    char\* revAlphabet = alphabet + 25;

    for (int i = 0; i < 26; i++) {

        printf("%c", \*(revAlphabet - i));

    }

    free(alphabet);

    return 0;

}

**Fixes Applied:**

1. **Initialized ptrName** with fullName before printing it.
2. **Fixed the typo \*ptrSallary to \*ptrSalary.**
3. **Checked for successful memory allocation** before dereferencing pointers.
4. **Avoided accessing freed memory** by resetting pointers to NULL after free().
5. **Corrected buffer allocation for alphabet** to accommodate 26 letters and a null terminator.
6. **Removed invalid pointer arithmetic** and ensured valid memory access.