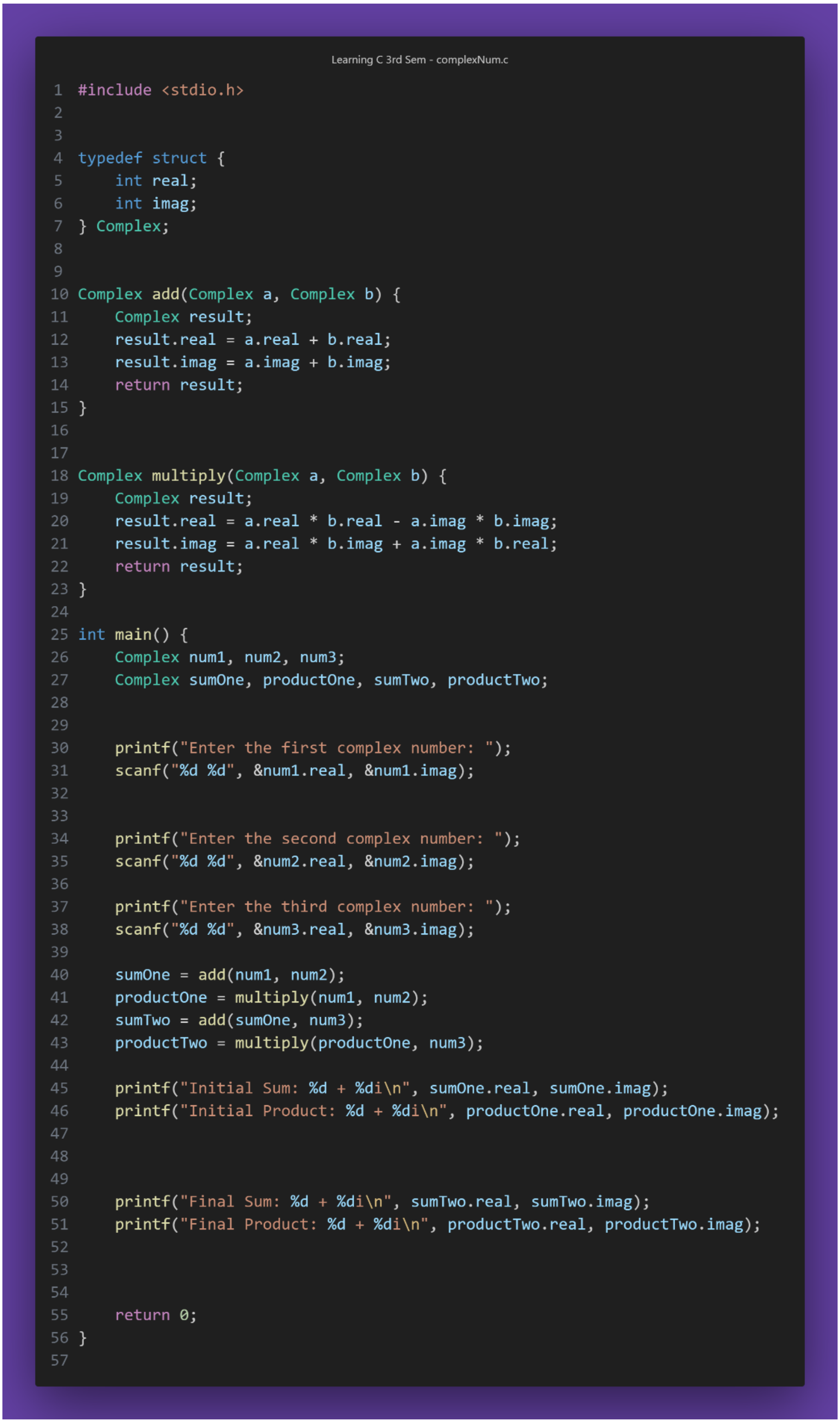
Name: Utkarsh Yadav

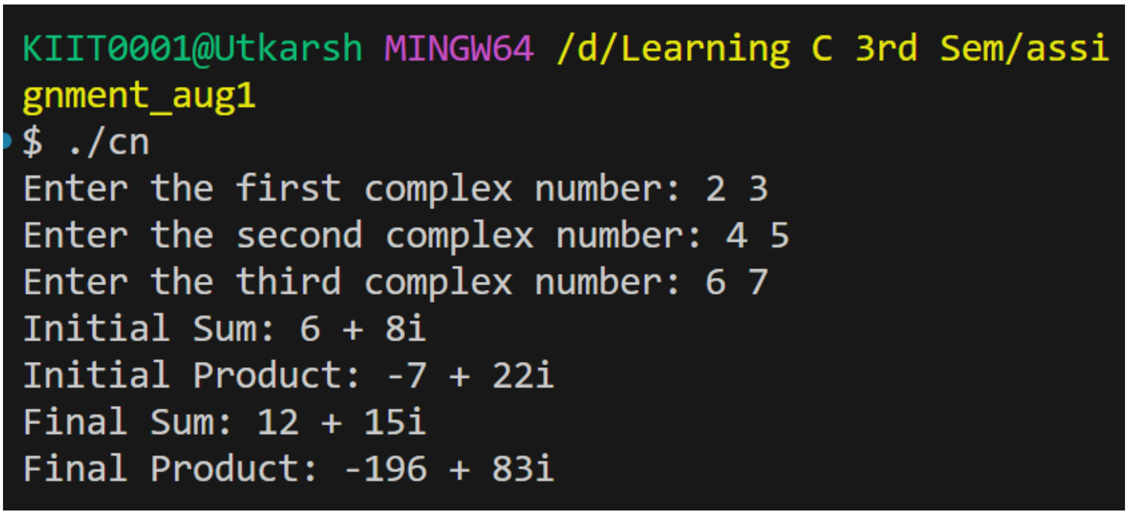
Roll: 23053172

DS Assignment for 01/08/2024

Add and multiply 3 complex numbers using the old function

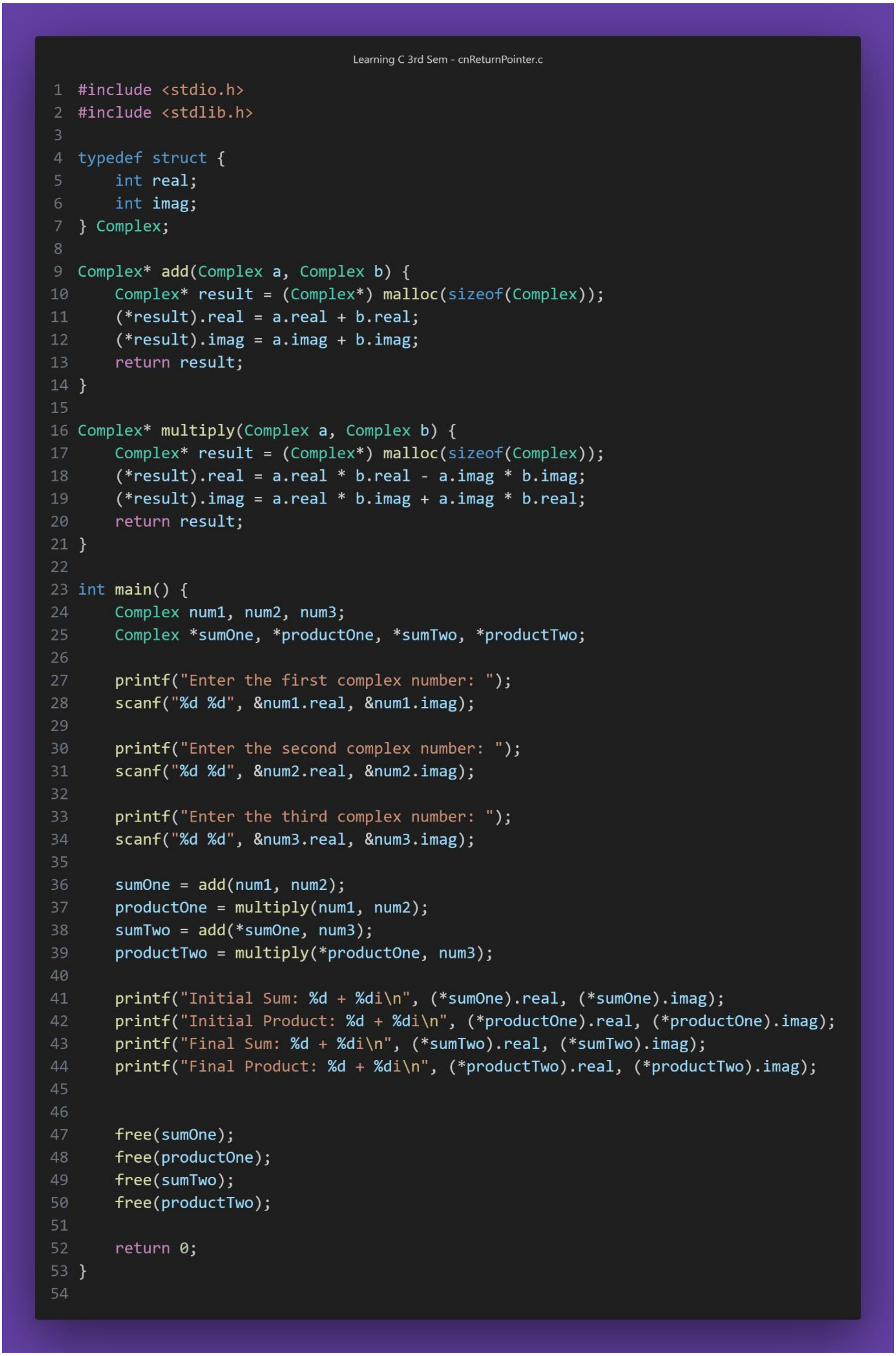


```c
#include <stdio.h>


typedef struct {
    int real;
    int imag;
} Complex;


Complex add(Complex a, Complex b) {
    Complex result;
    result.real = a.real + b.real;
    result.imag = a.imag + b.imag;
    return result;
}


Complex multiply(Complex a, Complex b) {
    Complex result;
    result.real = a.real * b.real - a.imag * b.imag;
    result.imag = a.real * b.imag + a.imag * b.real;
    return result;
}

int main() {
    Complex num1, num2, num3;
    Complex sumOne, productOne, sumTwo, productTwo;


    printf("Enter the first complex number: ");
    scanf("%d %d", &num1.real, &num1.imag);


    printf("Enter the second complex number: ");
    scanf("%d %d", &num2.real, &num2.imag);

    printf("Enter the third complex number: ");
    scanf("%d %d", &num3.real, &num3.imag);

    sumOne = add(num1, num2);
    productOne = multiply(num1, num2);
    sumTwo = add(sumOne, num3);
    productTwo = multiply(productOne, num3);

    printf("Initial Sum: %d + %di\n", sumOne.real, sumOne.imag);
    printf("Initial Product: %d + %di\n", productOne.real, productOne.imag);



    printf("Final Sum: %d + %di\n", sumTwo.real, sumTwo.imag);
    printf("Final Product: %d + %di\n", productTwo.real, productTwo.imag);



    return 0;
}

```

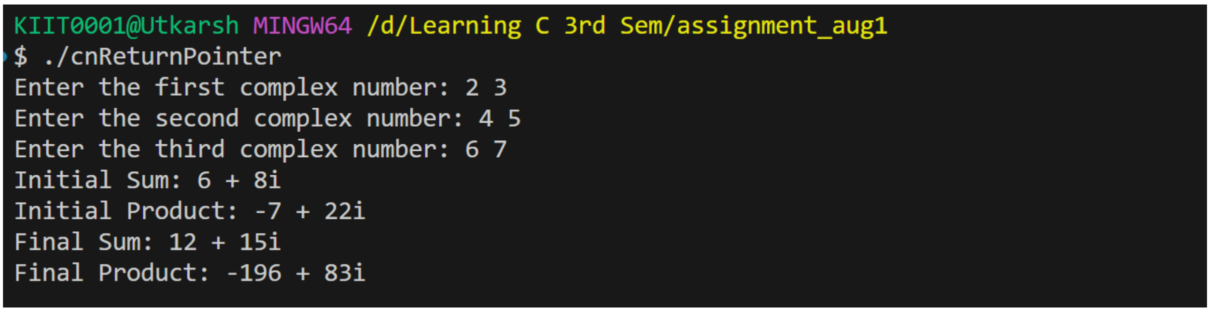


```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assi
gnment_aug1
$ ./cn
Enter the first complex number: 2 3
Enter the second complex number: 4 5
Enter the third complex number: 6 7
Initial Sum: 6 + 8i
Initial Product: -7 + 22i
Final Sum: 12 + 15i
Final Product: -196 + 83i
```

Return a struct pointer in the same program

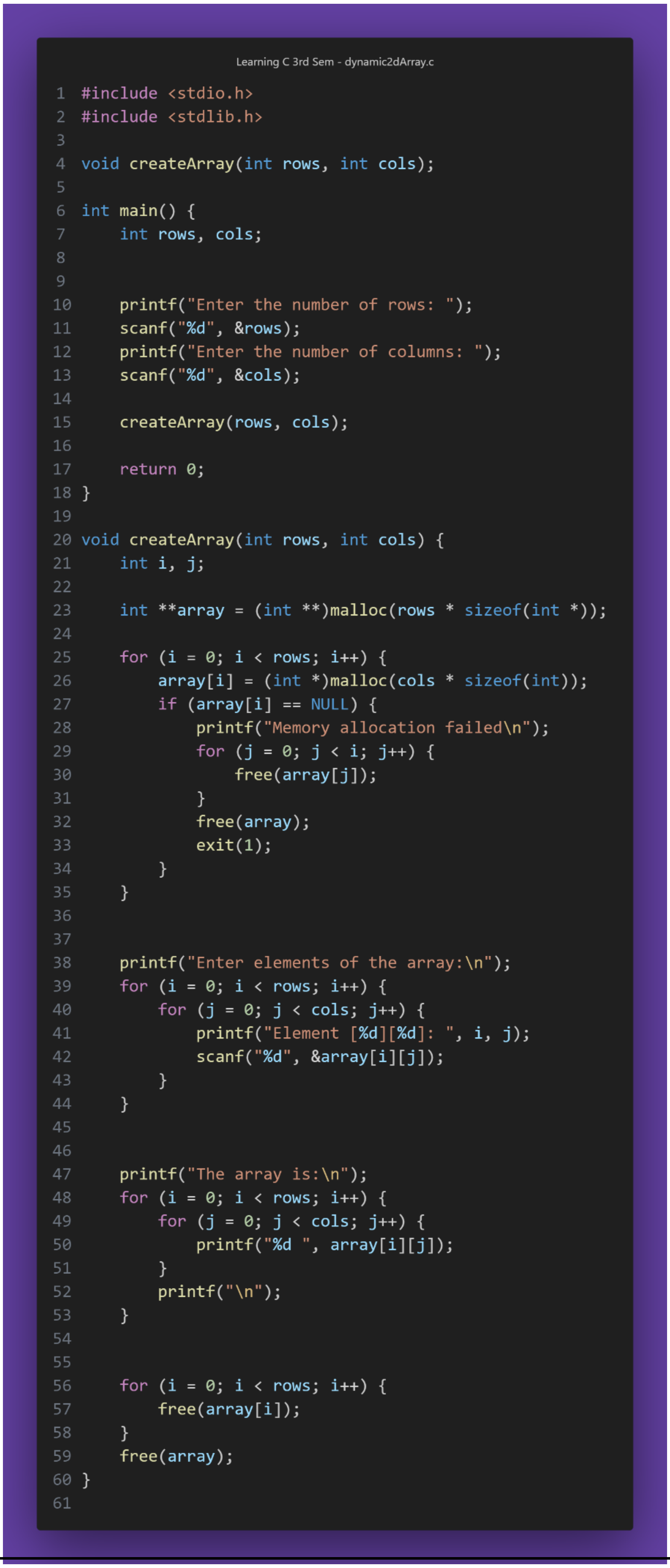


```c
#include <stdio.h>
#include <stdlib.h>

typedef struct {
    int real;
    int imag;
} Complex;

Complex* add(Complex a, Complex b) {
    Complex* result = (Complex*) malloc(sizeof(Complex));
    (*result).real = a.real + b.real;
    (*result).imag = a.imag + b.imag;
    return result;
}

Complex* multiply(Complex a, Complex b) {
    Complex* result = (Complex*) malloc(sizeof(Complex));
    (*result).real = a.real * b.real - a.imag * b.imag;
    (*result).imag = a.real * b.imag + a.imag * b.real;
    return result;
}

int main() {
    Complex num1, num2, num3;
    Complex *sumOne, *productOne, *sumTwo, *productTwo;

    printf("Enter the first complex number: ");
    scanf("%d %d", &num1.real, &num1.imag);

    printf("Enter the second complex number: ");
    scanf("%d %d", &num2.real, &num2.imag);

    printf("Enter the third complex number: ");
    scanf("%d %d", &num3.real, &num3.imag);

    sumOne = add(num1, num2);
    productOne = multiply(num1, num2);
    sumTwo = add(*sumOne, num3);
    productTwo = multiply(*productOne, num3);

    printf("Initial Sum: %d + %di\n", (*sumOne).real, (*sumOne).imag);
    printf("Initial Product: %d + %di\n", (*productOne).real, (*productOne).imag);
    printf("Final Sum: %d + %di\n", (*sumTwo).real, (*sumTwo).imag);
    printf("Final Product: %d + %di\n", (*productTwo).real, (*productTwo).imag);


    free(sumOne);
    free(productOne);
    free(sumTwo);
    free(productTwo);

    return 0;
}
```

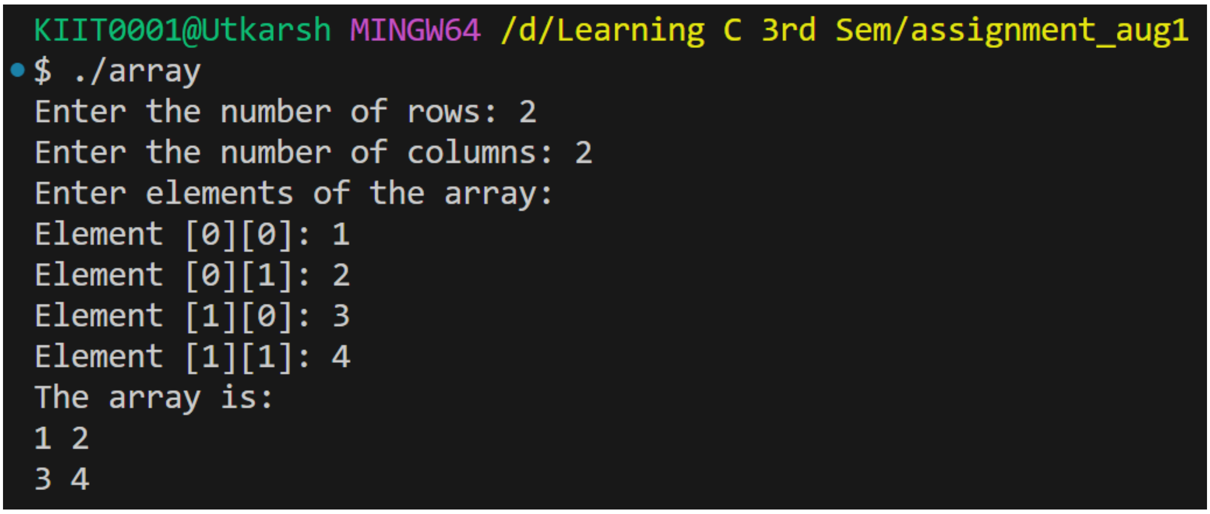


```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./cnReturnPointer
Enter the first complex number: 2 3
Enter the second complex number: 4 5
Enter the third complex number: 6 7
Initial Sum: 6 + 8i
Initial Product: -7 + 22i
Final Sum: 12 + 15i
Final Product: -196 + 83i
```

Dynamically allocate and display a 2D Array

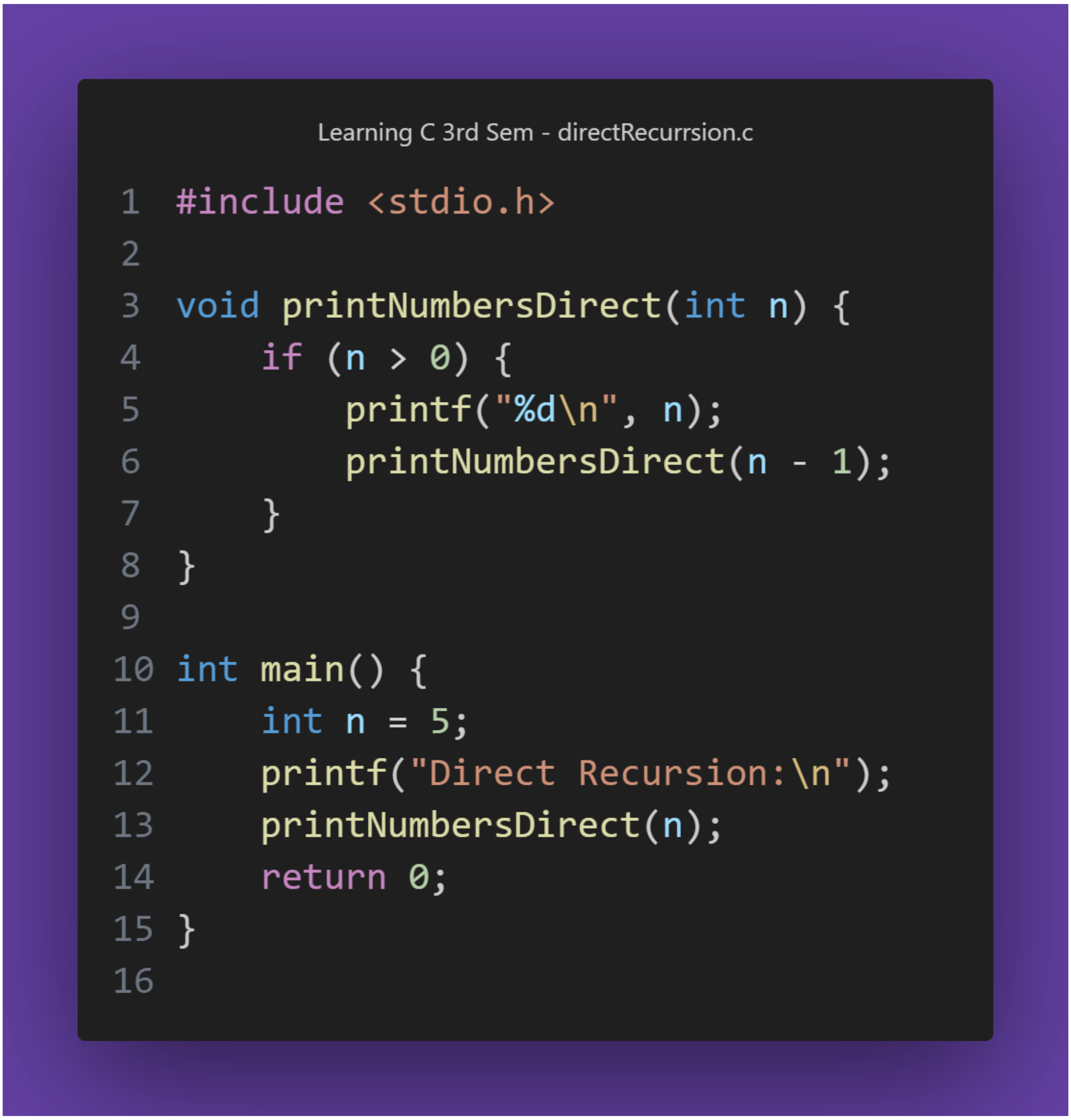


```c
#include <stdio.h>
#include <stdlib.h>

void createArray(int rows, int cols);

int main() {
    int rows, cols;


    printf("Enter the number of rows: ");
    scanf("%d", &rows);
    printf("Enter the number of columns: ");
    scanf("%d", &cols);

    createArray(rows, cols);

    return 0;
}

void createArray(int rows, int cols) {
    int i, j;

    int **array = (int **)malloc(rows * sizeof(int *));

    for (i = 0; i < rows; i++) {
        array[i] = (int *)malloc(cols * sizeof(int));
        if (array[i] == NULL) {
            printf("Memory allocation failed\n");
            for (j = 0; j < i; j++) {
                free(array[j]);
            }
            free(array);
            exit(1);
        }
    }


    printf("Enter elements of the array:\n");
    for (i = 0; i < rows; i++) {
        for (j = 0; j < cols; j++) {
            printf("Element [%d][%d]: ", i, j);
            scanf("%d", &array[i][j]);
        }
    }


    printf("The array is:\n");
    for (i = 0; i < rows; i++) {
        for (j = 0; j < cols; j++) {
            printf("%d ", array[i][j]);
        }
        printf("\n");
    }


    for (i = 0; i < rows; i++) {
        free(array[i]);
    }
    free(array);
}

```



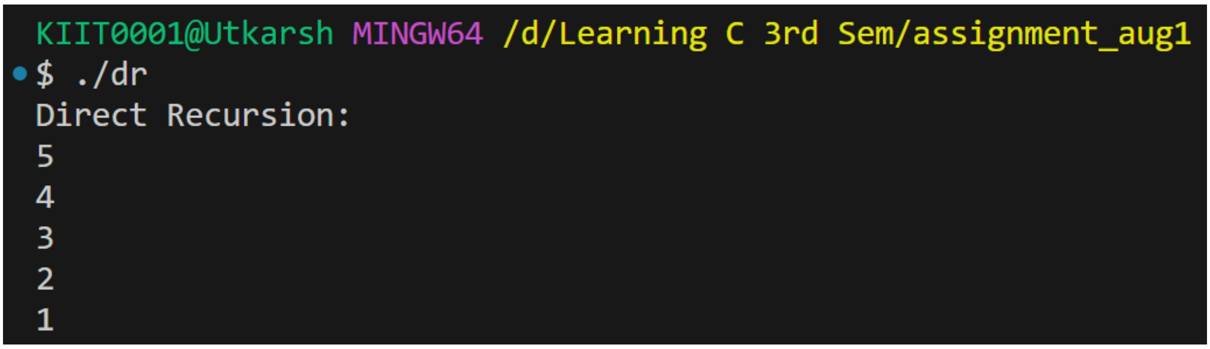
```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./array
Enter the number of rows: 2
Enter the number of columns: 2
Enter elements of the array:
Element [0][0]: 1
Element [0][1]: 2
Element [1][0]: 3
Element [1][1]: 4
The array is:
1 2
3 4
```

Example of Direct Recursion



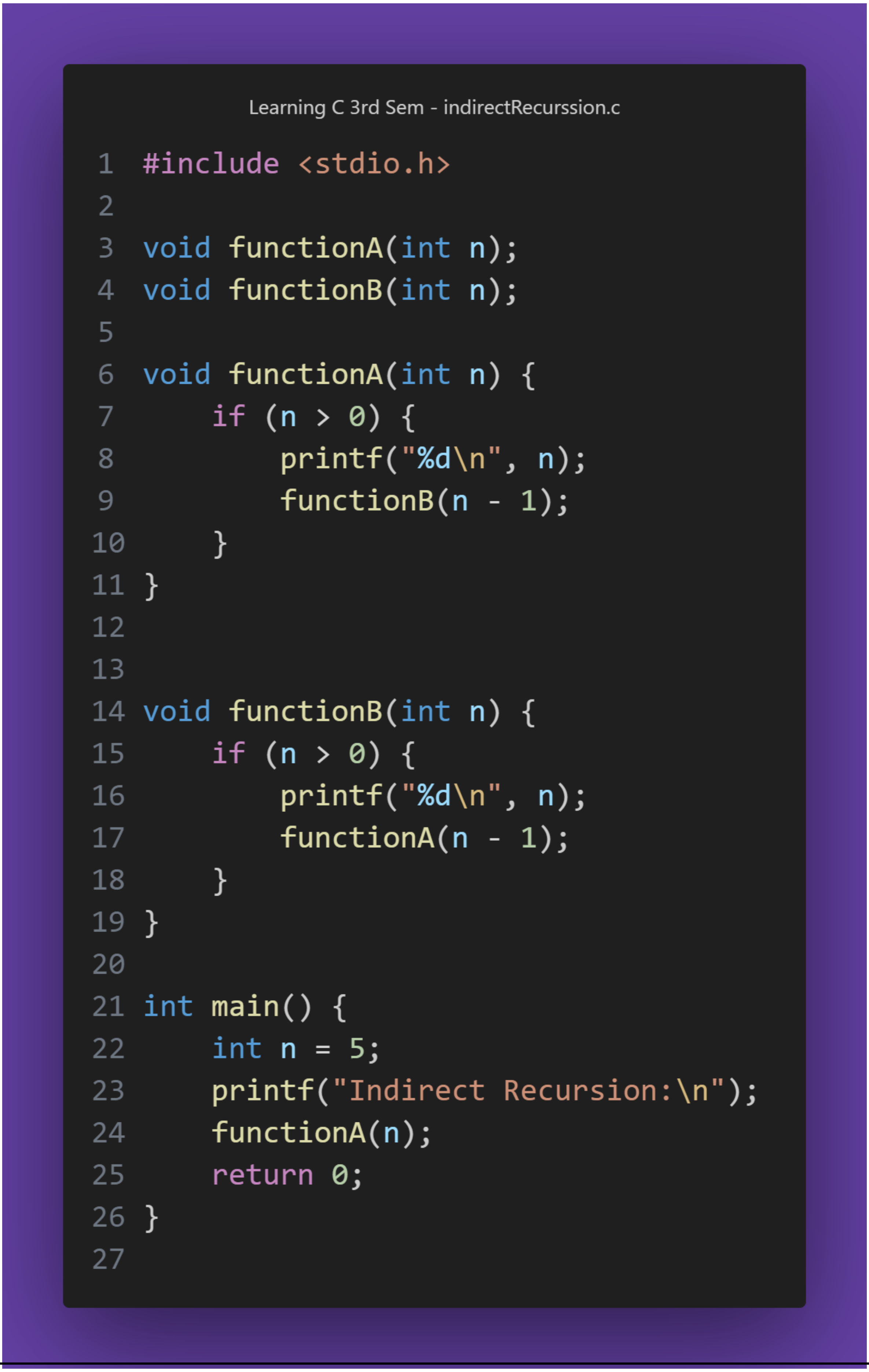
Learning C 3rd Sem - directRecurrsion.c

```c
#include <stdio.h>

void printNumbersDirect(int n) {
    if (n > 0) {
        printf("%d\n", n);
        printNumbersDirect(n - 1);
    }
}

int main() {
    int n = 5;
    printf("Direct Recursion:\n");
    printNumbersDirect(n);
    return 0;
}

```



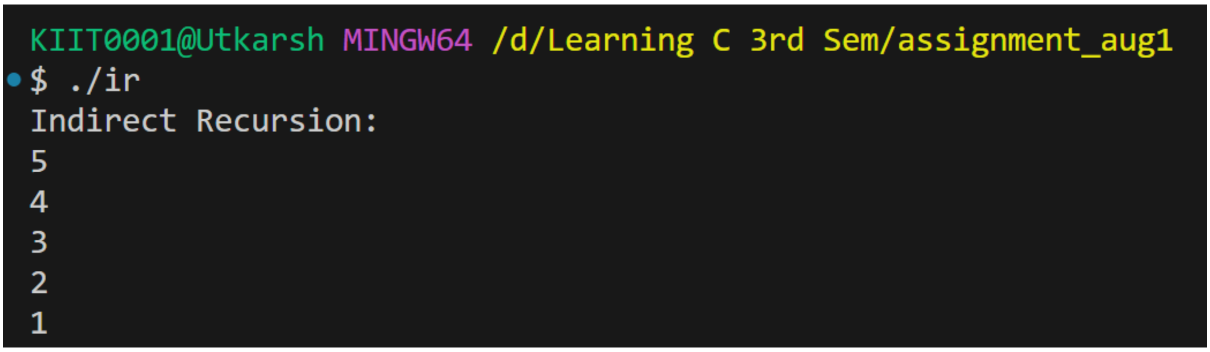
```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./dr
Direct Recursion:
5
4
3
2
1
```

Example of Indirect Recursion



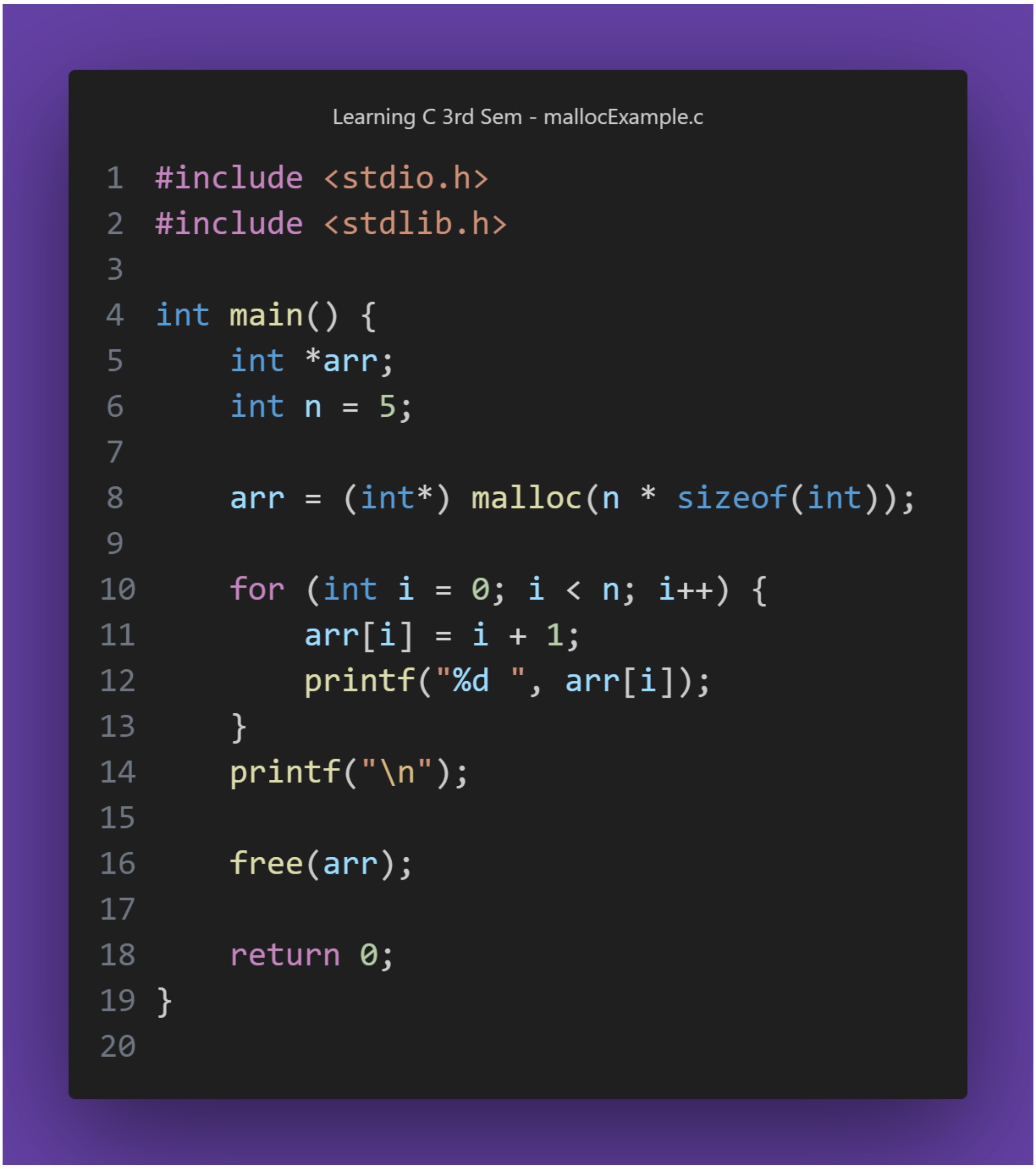
Learning C 3rd Sem - indirectRecurssion.c

```c
#include <stdio.h>

void functionA(int n);
void functionB(int n);

void functionA(int n) {
    if (n > 0) {
        printf("%d\n", n);
        functionB(n - 1);
    }
}


void functionB(int n) {
    if (n > 0) {
        printf("%d\n", n);
        functionA(n - 1);
    }
}

int main() {
    int n = 5;
    printf("Indirect Recursion:\n");
    functionA(n);
    return 0;
}
```



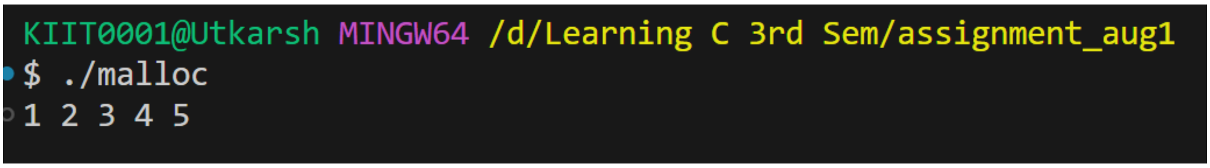
```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./ir
Indirect Recursion:
5
4
3
2
1
```

Malloc Example



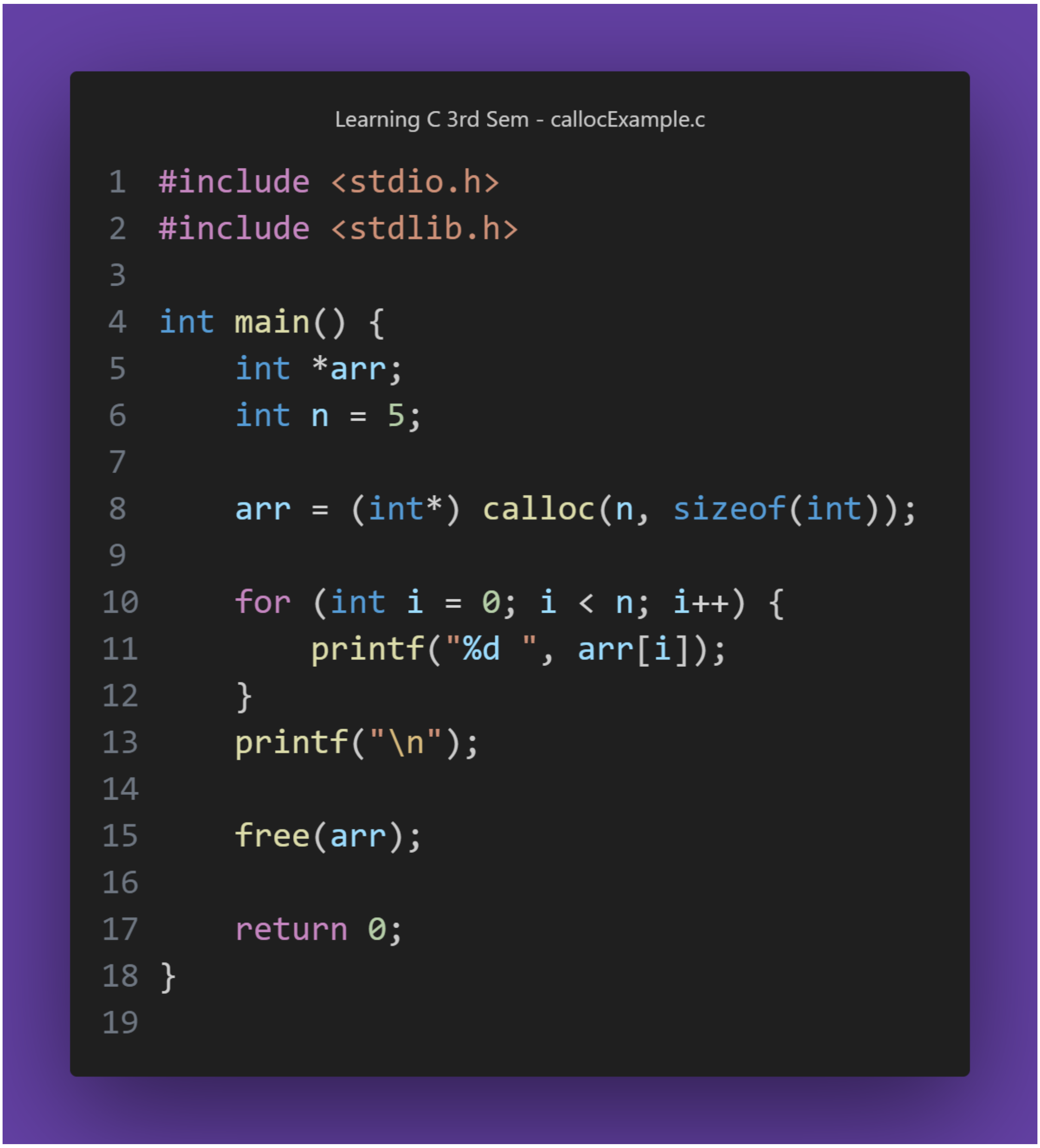
Learning C 3rd Sem - mallocExample.c

```c
#include <stdio.h>
#include <stdlib.h>

int main() {
    int *arr;
    int n = 5;

    arr = (int*) malloc(n * sizeof(int));

    for (int i = 0; i < n; i++) {
        arr[i] = i + 1;
        printf("%d ", arr[i]);
    }
    printf("\n");

    free(arr);

    return 0;
}

```



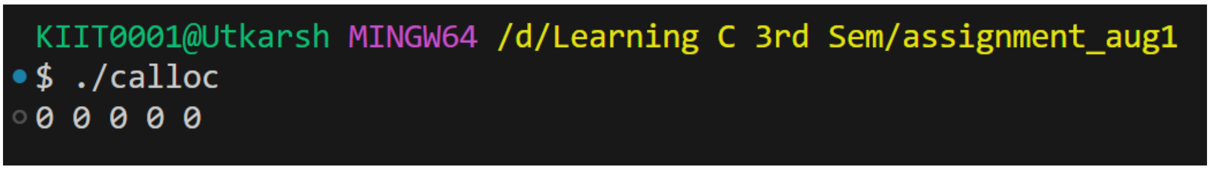
```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./malloc
1 2 3 4 5
```

Calloc Example



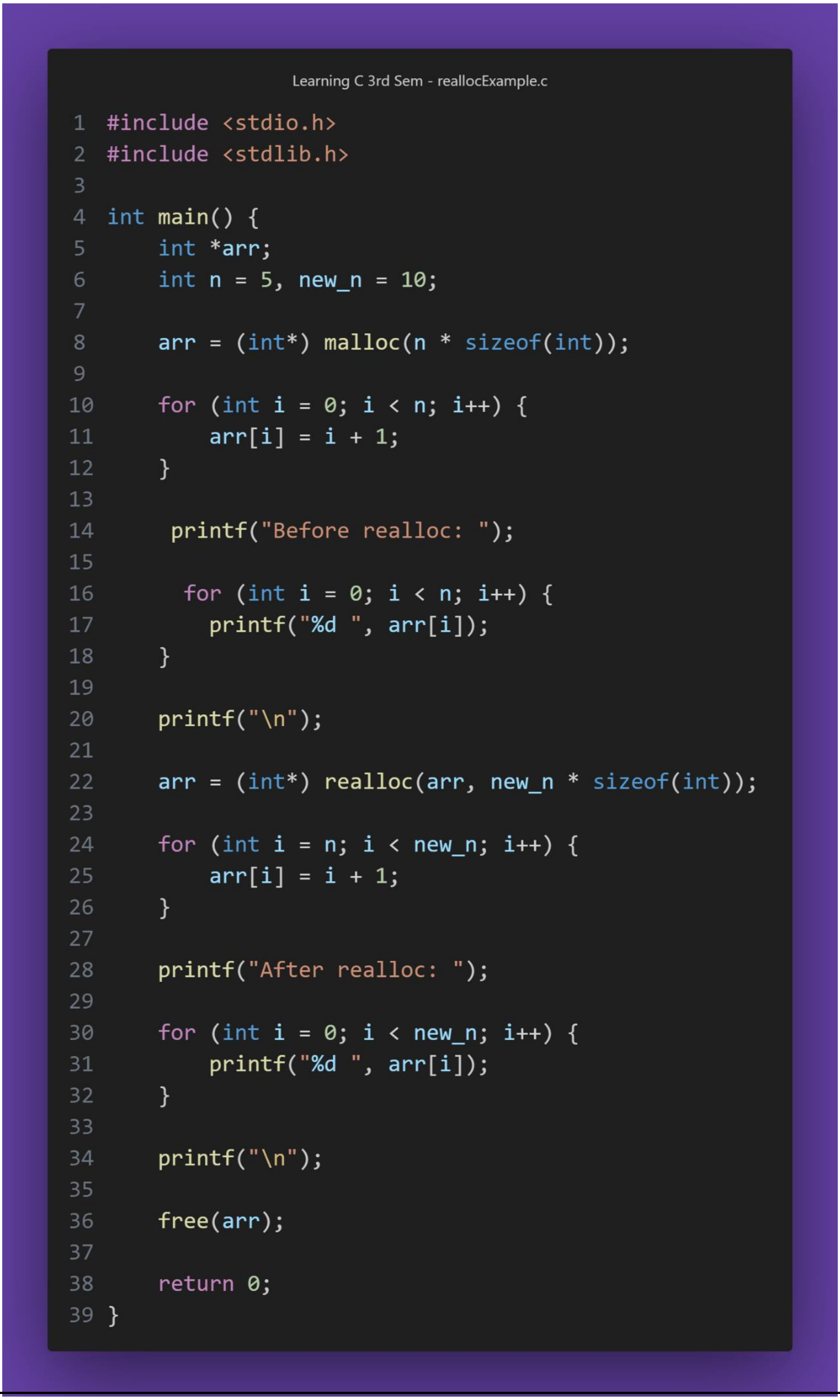
Learning C 3rd Sem - callocExample.c

```c
#include <stdio.h>
#include <stdlib.h>

int main() {
    int *arr;
    int n = 5;

    arr = (int*) calloc(n, sizeof(int));

    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    free(arr);

    return 0;
}

```



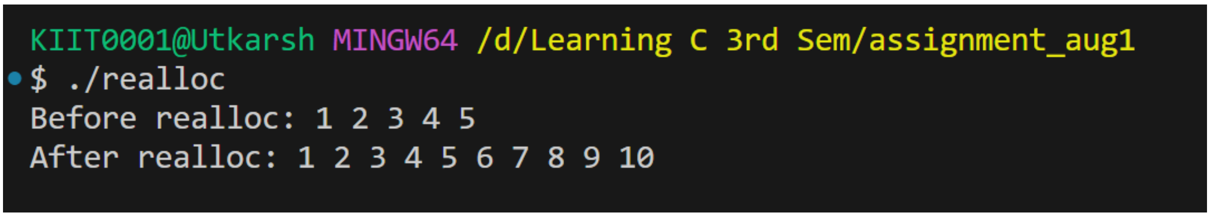
```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./calloc
0 0 0 0 0
```

Realloc Example



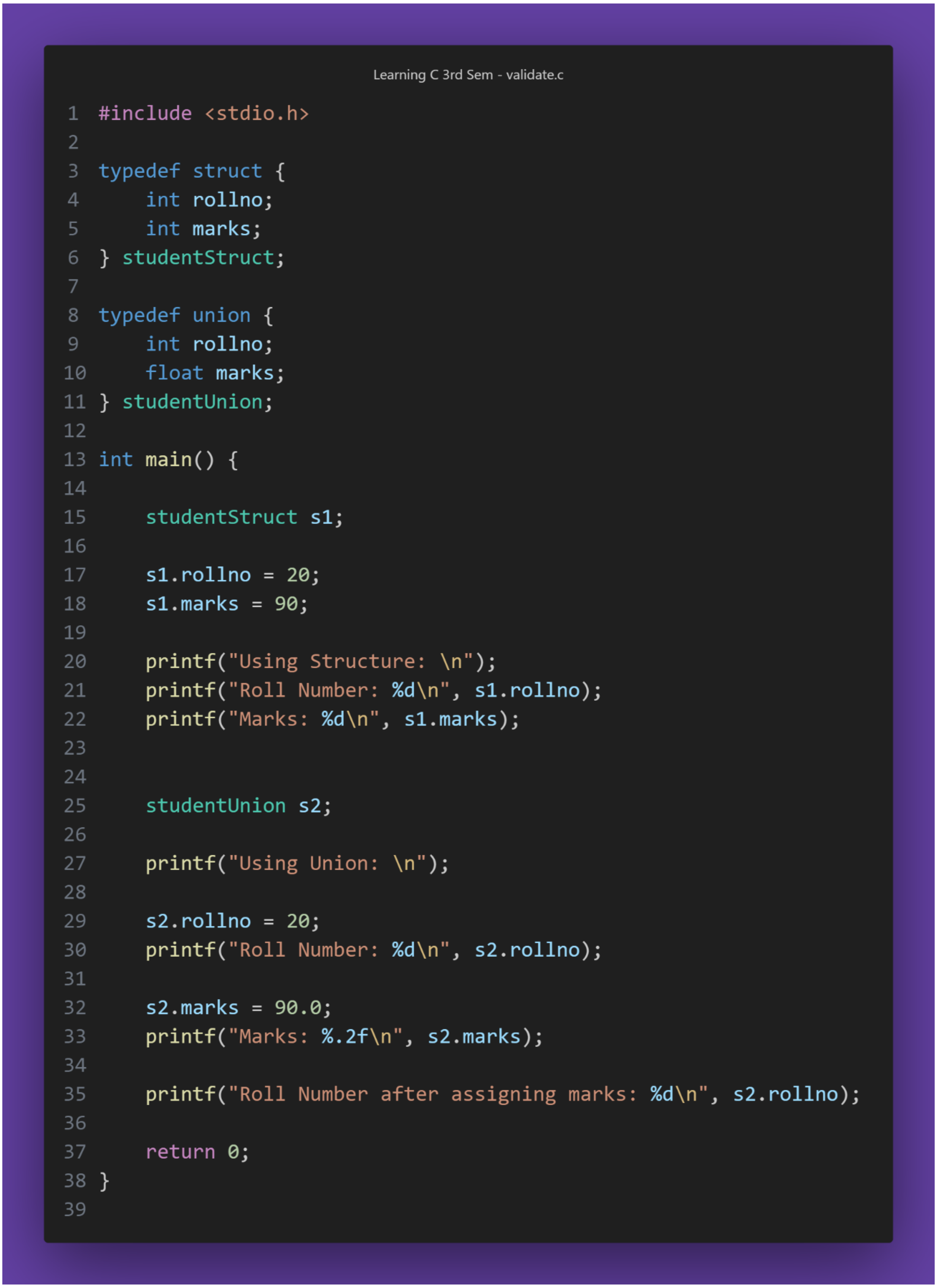
Learning C 3rd Sem - reallocExample.c

```c
#include <stdio.h>
#include <stdlib.h>

int main() {
    int *arr;
    int n = 5, new_n = 10;

    arr = (int*) malloc(n * sizeof(int));

    for (int i = 0; i < n; i++) {
        arr[i] = i + 1;
    }

     printf("Before realloc: ");

      for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }

    printf("\n");

    arr = (int*) realloc(arr, new_n * sizeof(int));

    for (int i = n; i < new_n; i++) {
        arr[i] = i + 1;
    }

    printf("After realloc: ");

    for (int i = 0; i < new_n; i++) {
        printf("%d ", arr[i]);
    }

    printf("\n");

    free(arr);

    return 0;
}
```

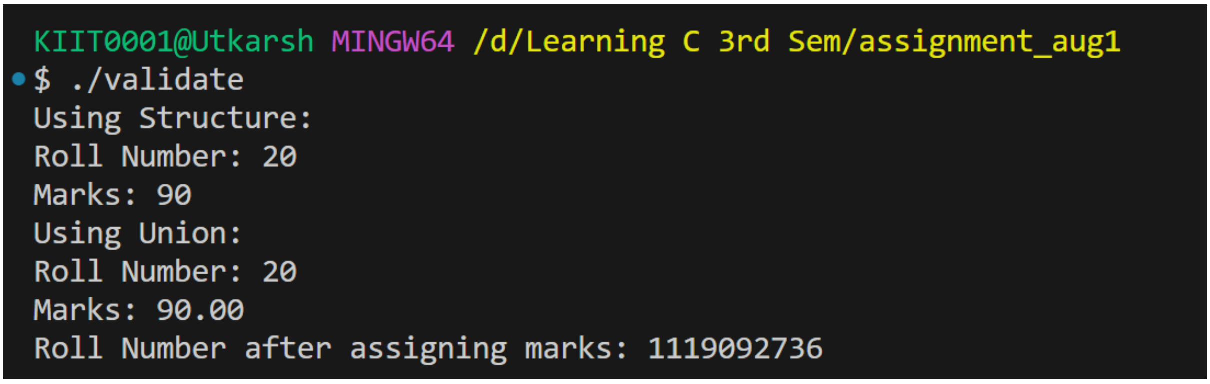


```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./realloc
Before realloc: 1 2 3 4 5
After realloc: 1 2 3 4 5 6 7 8 9 10
```

Validate the structure and union output from the ppt

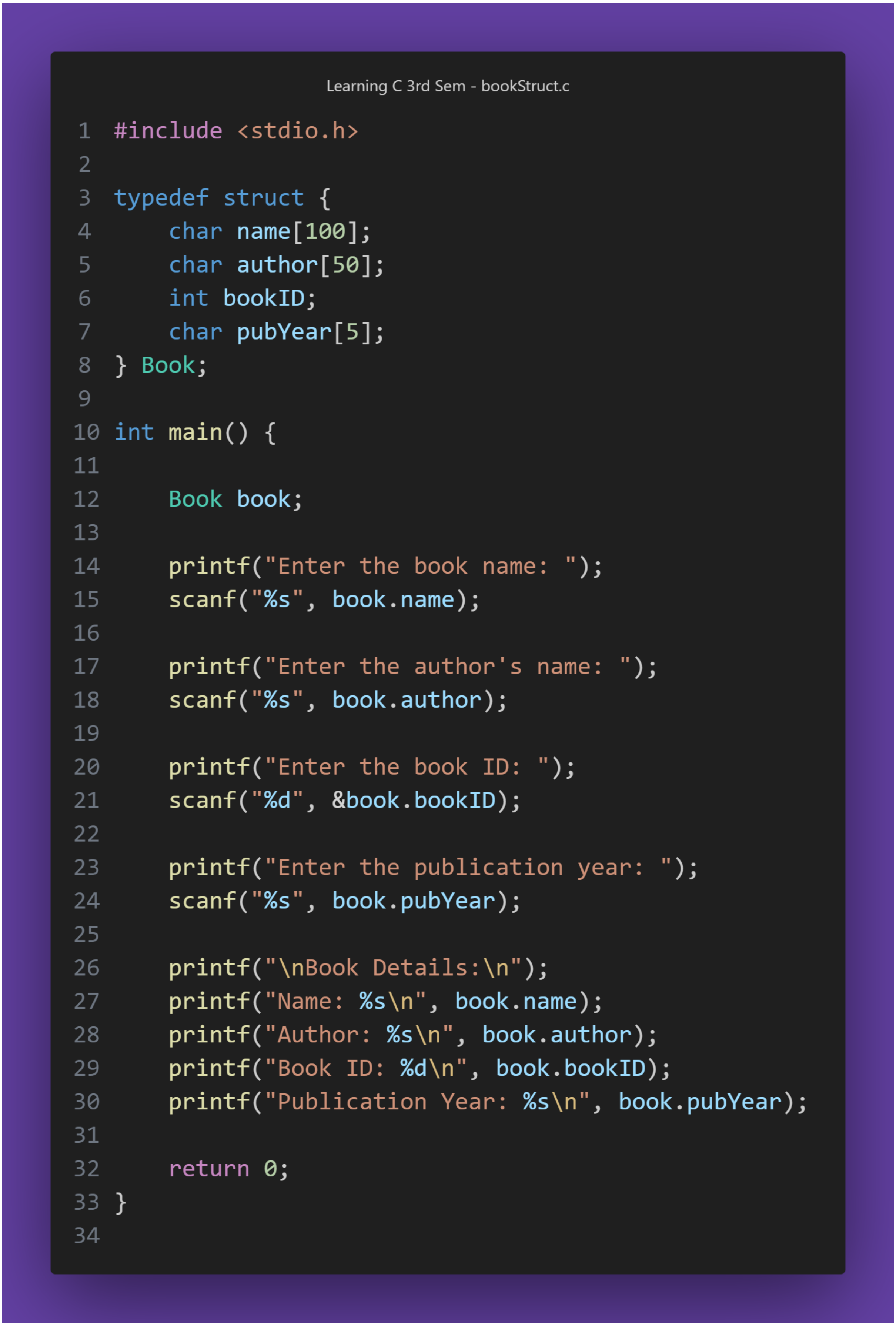


```c
#include <stdio.h>

typedef struct {
    int rollno;
    int marks;
} studentStruct;

typedef union {
    int rollno;
    float marks;
} studentUnion;

int main() {

    studentStruct s1;

    s1.rollno = 20;
    s1.marks = 90;

    printf("Using Structure: \n");
    printf("Roll Number: %d\n", s1.rollno);
    printf("Marks: %d\n", s1.marks);


    studentUnion s2;

    printf("Using Union: \n");

    s2.rollno = 20;
    printf("Roll Number: %d\n", s2.rollno);

    s2.marks = 90.0;
    printf("Marks: %.2f\n", s2.marks);

    printf("Roll Number after assigning marks: %d\n", s2.rollno);

    return 0;
}

```

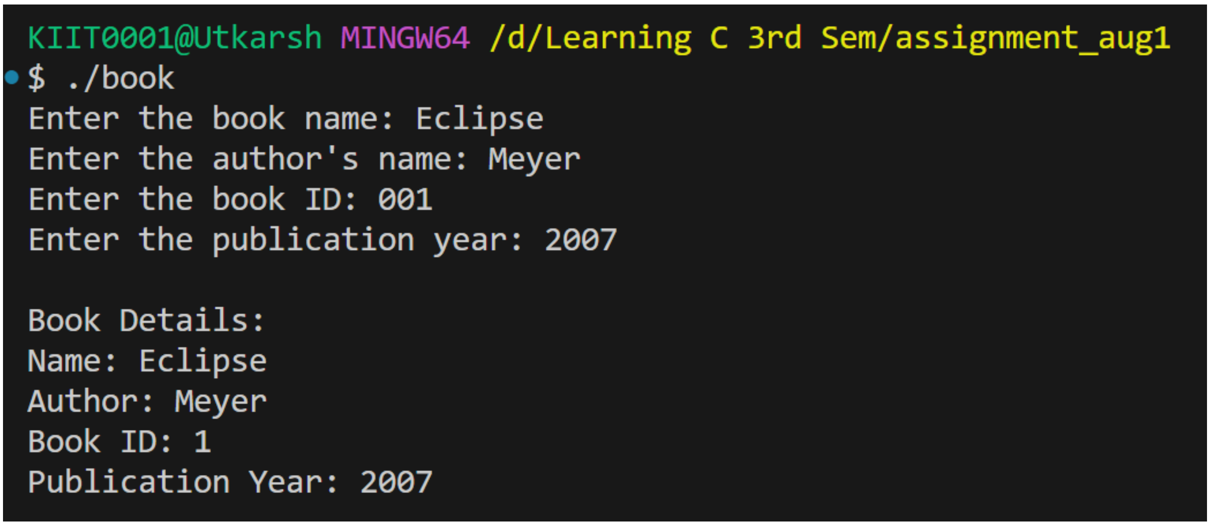


```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./validate
Using Structure:
Roll Number: 20
Marks: 90
Using Union:
Roll Number: 20
Marks: 90.00
Roll Number after assigning marks: 1119092736
```

Struct to store and display Book details



```c
#include <stdio.h>

typedef struct {
    char name[100];
    char author[50];
    int bookID;
    char pubYear[5];
} Book;

int main() {

    Book book;

    printf("Enter the book name: ");
    scanf("%s", book.name);

    printf("Enter the author's name: ");
    scanf("%s", book.author);

    printf("Enter the book ID: ");
    scanf("%d", &book.bookID);

    printf("Enter the publication year: ");
    scanf("%s", book.pubYear);

    printf("\nBook Details:\n");
    printf("Name: %s\n", book.name);
    printf("Author: %s\n", book.author);
    printf("Book ID: %d\n", book.bookID);
    printf("Publication Year: %s\n", book.pubYear);

    return 0;
}
```



```
KIIT0001@Utkarsh MINGW64 /d/Learning C 3rd Sem/assignment_aug1
$ ./book
Enter the book name: Eclipse
Enter the author's name: Meyer
Enter the book ID: 001
Enter the publication year: 2007

Book Details:
Name: Eclipse
Author: Meyer
Book ID: 1
Publication Year: 2007
```