Day 1 : Cprograms Assignment:

1. Write a C program to create an integer array of size 5, initialize it with values from 1 to 5, and then use pointer arithmetic to print each element of the array.

#include <stdio.h>

int main()

{

int array[5] = {1, 2, 3, 4, 5};

int \*ptr = array;

printf("Array elements: \n");

for (int i = 0; i < 5; i++)

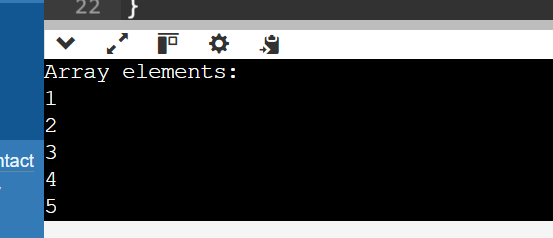
{

printf("%d\n", \*(ptr + i));

}

return 0;

}



1. Write a C program to create a pointer to a pointer for an integer variable. Initialize the integer variable with a value, and then print its value using both the single pointer and the pointer to pointer.

#include <stdio.h>

int main()

{

int x = 25;

int \*ptr1 = &x;

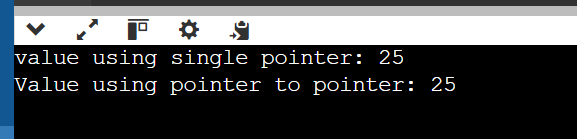
int \*\*ptr2 = &ptr1;

printf("value using single pointer: %d\n", \*ptr1);

printf("Value using pointer to pointer: %d\n", \*\*ptr2);

return 0;

}



1. Write a C function void swap(int \*a, int \*b) that swaps the values of two integers. Then, write a main function to test this swap function using pointer arguments.

#include <stdio.h>

void swap(int \*a, int \*b)

{

int temp = \*a;

\*a = \*b;

\*b = temp;

}

int main()

{

int p = 10;

int q = 20;

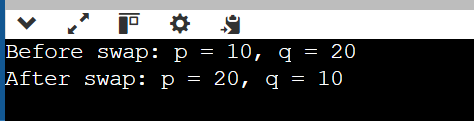
printf("Before swap: p = %d, q = %d\n", p, q);

swap(&p, &q);

printf("After swap: p = %d, q = %d\n", p, q);

return 0;

}



1. Write a C program to create a function pointer that points to a function int add(int, int). Use the function pointer to call the add function and print the result.

#include <stdio.h>

int add(int a, int b)

{

return a + b;

}

int main()

{

int (\*func\_ptr)(int, int);

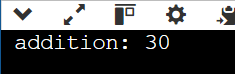
func\_ptr = &add;

int result = func\_ptr(10,20);

printf(" addition: %d\n", result);

return 0;

}



1. Write a C function int factorial(int n) that calculates the factorial of a given number using recursion. Test this function in the main program by calculating and printing the factorial of 5.

#include <stdio.h>

int factorial(int n)

{

if (n <= 1)

{

return 1;

}

else

{

return n \* factorial(n - 1);

}

}

int main()

{

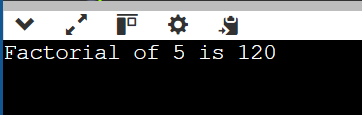
int num = 5;

int result = factorial(num);

printf("Factorial of %d is %d\n", num, result);

return 0;

}



1. Write a C program to create an array of function pointers, where each function takes two integers as arguments and returns an integer. Include functions for addition, subtraction, multiplication, and division. Use the array to perform these operations on two integers and print the results.

#include <stdio.h>

int add(int a, int b) {

return a + b;

}

int subtract(int a, int b) {

return a - b;

}

int multiply(int a, int b) {

return a \* b;

}

int divide(int a, int b) {

if (b != 0) {

return a / b;

} else {

printf("Error: Division by zero\n");

return 0;

}

}

int main() {

int (\*operations[4])(int, int) = {add, subtract, multiply, divide};

int x = 22;

int y = 33;

printf("Addition of %d and %d: %d\n", x, y, operations[0](x, y));

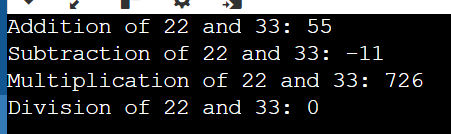
printf("Subtraction of %d and %d: %d\n", x, y, operations[1](x, y));

printf("Multiplication of %d and %d: %d\n", x, y, operations[2](x, y));

printf("Division of %d and %d: %d\n", x, y, operations[3](x, y));

return 0;

}



1. Define a structure struct Point with two integer members x and y. Write a C program to create a Point variable, initialize it with values, and print the values.

#include <stdio.h>

struct Point {

int x;

int y;

};

int main() {

struct Point p = {10, 20};

printf("Point variables:\n");

printf("x: %d\n", p.x);

printf("y: %d\n", p.y);

return 0;

}



1. Write a C program to define a structure struct Student with members name, age, and marks. Create an array of 3 students, initialize them with values, and print the details of each student.

#include <stdio.h>

struct Student {

char name[50];

int age;

float marks;

};

int main() {

struct Student students[3] = {

{"Vidya", 20, 85.5},

{"kavya", 21, 90.0},

{"divya", 22, 95.5}

};

for(int i = 0; i < 3; i++) {

printf("Student %d:\n", i + 1);

printf("Name: %s\n", students[i].name);

printf("Age: %d\n", students[i].age);

printf("Marks: %.2f\n", students[i].marks);

printf("\n");

}

return 0;

} 