PF LAB 10:

Q1:

#include <stdio.h>

void printArrayRecursive(int arr[], int size) {

if (size == 0) return;

printf("%d ", arr[0]);

printArrayRecursive(arr + 1, size - 1);

}

int main() {

int n;

printf("Enter the size of the array: ");

scanf("%d", &n);

int arr[n];

printf("Enter %d elements: ", n);

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

scanf("%d", &arr[i]);

}

printf("Array elements: ");

printArrayRecursive(arr, n);

printf("\n");

return 0;

}

Q2:  
#include <stdio.h>

#include <math.h>

struct Point {

int x;

int y;

};

double calculateDistance(struct Point p1, struct Point p2) {

return sqrt((p2.x - p1.x) \* (p2.x - p1.x) + (p2.y - p1.y) \* (p2.y - p1.y));

}

int isPointInRectangle(struct Point p, struct Point bottomLeft, struct Point topRight) {

return p.x >= bottomLeft.x && p.x <= topRight.x && p.y >= bottomLeft.y && p.y <= topRight.y;

}

int main() {

struct Point p1, p2, bottomLeft, topRight;

printf("Enter coordinates of first point (x y): ");

scanf("%d %d", &p1.x, &p1.y);

printf("Enter coordinates of second point (x y): ");

scanf("%d %d", &p2.x, &p2.y);

double distance = calculateDistance(p1, p2);

printf("Distance between the two points: %.2f\n", distance);

printf("Enter bottom-left coordinates of the rectangle (x y): ");

scanf("%d %d", &bottomLeft.x, &bottomLeft.y);

printf("Enter top-right coordinates of the rectangle (x y): ");

scanf("%d %d", &topRight.x, &topRight.y);

printf("Enter point coordinates to check (x y): ");

struct Point checkPoint;

scanf("%d %d", &checkPoint.x, &checkPoint.y);

if (isPointInRectangle(checkPoint, bottomLeft, topRight)) {

printf("The point is inside the rectangle.\n");

} else {

printf("The point is outside the rectangle.\n") }

return 0;

}

Q3:

#include <stdio.h>

#define MAX\_TEMP 40

void checkTemperature(int temp) {

static int count = 0;

if (temp > MAX\_TEMP) {

count++;

printf("Temperature exceeded the limit! Exceeded count: %d\n", count);

} else {

printf("Temperature is within the limit.\n");

}

}

int main() {

int temp;

char choice;

do {

printf("Enter the temperature (Celsius): ");

scanf("%d", &temp);

checkTemperature(temp);

printf("Do you want to enter another temperature? (y/n): ");

scanf(" %c", &choice);

} while (choice == 'y' || choice == 'Y');

return 0;

}

Q4:

#include <stdio.h>

#include <string.h>

#define MAX\_CARS 100

struct Car {

char make[30];

char model[30];

int year;

float price;

float mileage;

};

void addCar(struct Car cars[], int \*count) {

printf("Enter make: ");

scanf("%s", cars[\*count].make);

printf("Enter model: ");

scanf("%s", cars[\*count].model);

printf("Enter year: ");

scanf("%d", &cars[\*count].year);

printf("Enter price: ");

scanf("%f", &cars[\*count].price);

printf("Enter mileage: ");

scanf("%f", &cars[\*count].mileage);

(\*count)++;

}

void displayCars(struct Car cars[], int count) {

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

printf("Make: %s, Model: %s, Year: %d, Price: %.2f, Mileage: %.2f\n", cars[i].make, cars[i].model, cars[i].year, cars[i].price, cars[i].mileage);

}

}

void searchCars(struct Car cars[], int count, char searchTerm[]) {

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

if (strstr(cars[i].make, searchTerm) != NULL || strstr(cars[i].model, searchTerm) != NULL) {

printf("Make: %s, Model: %s, Year: %d, Price: %.2f, Mileage: %.2f\n", cars[i].make, cars[i].model, cars[i].year, cars[i].price, cars[i].mileage);

}

}

}

int main() {

struct Car cars[MAX\_CARS];

int count = 0;

int choice;

do {

printf("1. Add car\n2. Display cars\n3. Search cars\n4. Exit\nEnter choice: ");

scanf("%d", &choice);

if (choice == 1) {

addCar(cars, &count);

} else if (choice == 2) {

displayCars(cars, count);

} else if (choice == 3) {

char searchTerm[30];

printf("Enter make or model to search: ");

scanf("%s", searchTerm);

searchCars(cars, count, searchTerm);

}

} while (choice != 4);

return 0;

}

Q5:

#include <stdio.h>

void bubbleSort(int arr[], int size) {

if (size == 1) return;

for (int i = 0; i < size - 1; i++) {

if (arr[i] > arr[i + 1]) {

int temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

}

}

bubbleSort(arr, size - 1);

}

int main() {

int arr[] = {64, 34, 25, 12, 22, 11, 90};

int size = sizeof(arr) / sizeof(arr[0]);

bubbleSort(arr, size);

printf("Sorted array: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q6:

#include <stdio.h>

void bubbleSort(int arr[], int size) {

if (size == 1) return;

for (int i = 0; i < size - 1; i++) {

if (arr[i] > arr[i + 1]) {

int temp = arr[i];

arr[i] = arr[i + 1];

arr[i + 1] = temp;

}

}

bubbleSort(arr, size - 1);

}

int main() {

int arr[] = {64, 34, 25, 12, 22, 11, 90};

int size = sizeof(arr) / sizeof(arr[0]);

bubbleSort(arr, size);

printf("Sorted array: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q7:

#include <stdio.h>

#define METER\_TO\_KM 0.001

double convertMetersToKilometers(double meters) {

static int callCount = 0;

callCount++;

printf("The function has been called %d times.\n", callCount);

return meters \* METER\_TO\_KM;

}

int main() {

double meters;

printf("Enter distance in meters: ");

scanf("%lf", &meters);

double kilometers = convertMetersToKilometers(meters);

printf("%.2f meters is equal to %.2f kilometers.\n", meters, kilometers);

printf("\nCalling the conversion function again...\n");

kilometers = convertMetersToKilometers(meters);

return 0;

}

Q8:

#include <stdio.h>

int linearSearch(int arr[], int size, int target, int currentIndex) {

if (currentIndex == size) {

return -1;

}

if (arr[currentIndex] == target) {

return currentIndex;

}

return linearSearch(arr, size, target, currentIndex + 1);

}

int main() {

int arr[] = {2, 4, 6, 8, 10, 12, 14};

int size = sizeof(arr) / sizeof(arr[0]);

int target;

printf("Enter the target element to search for: ");

scanf("%d", &target);

int result = linearSearch(arr, size, target, 0);

if (result != -1) {

printf("Element %d found at index %d.\n", target, result);

} else {

printf("Element %d not found in the array.\n", target);

}

return 0;

}