#include <stdio.h>

#include <stdlib.h>

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

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

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

}

printf("\n");

}

int main() {

int \*arr;

int size, i;

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

scanf("%d", &size);

arr = (int \*)malloc(size \* sizeof(int));

if (arr == NULL) {

printf("Memory allocation failed!\n");

return 1;

}

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

arr[i] = i \* 2;

printArray(arr, size);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

int main() {

int \*arr1, \*arr2;

int size, i;

printf("Enter size of array: ");

scanf("%d", &size);

arr1 = (int \*)malloc(size \* sizeof(int));

arr2 = (int \*)calloc(size, sizeof(int));

if (!arr1 || !arr2) {

printf("Memory allocation failed!\n");

}

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

arr1[i] = i + 1;

arr2[i] = i \* 2;

}

printf("Malloc Array: ");

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

printf("%d ", arr1[i]);

printf("\nCalloc Array: ");

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

printf("%d ", arr2[i]);

return 0;

}

#include <stdio.h>

int isPrime(int n) {

if (n <= 1)

return 0;

for (int i = 2; i < n / 2; i++)

if (n % i == 0)

return 0;

}

return 1;

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isPrime(num))

printf("%d is a prime number.\n", num);

else

printf("%d is not a prime number.\n", num);

return 0;

}

#include <stdio.h>

void printTree(int n) {

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= n - i; j++)

printf(" ");

for (int k = 1; k <= (2 \* i - 1); k++)

printf("\*");

printf("\n");

}

for (int i = 1; i < n / 2; i++) {

for (int j = 1; j < n - 1; j++)

printf(" ");

printf("|\n");

}

}

int main() {

int height;

printf("Enter tree height: ");

scanf("%d", &height);

if (height <= 0) {

printf("Invalid height! Enter a positive number.\n");

return 1;

}

printTree(height);

return 0;

}

#include <stdio.h>

void printNumbers(int n) {

for (int i = 1; i <= n; i++)

printf("%d ", i);

printf("\n");

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (num <= 0) {

printf("Invalid input! Enter a positive number.\n");

return 1;

}

printNumbers(num);

return 0;

}

Expected output: 1 2 3 4 5

#include <stdio.h>

void printPattern(int n) {

for (int i = 1; i <= n; i++) {

for (int j = 1; j <= i; j++)

printf("\* ");

}

printf("\n");

}

int main() {

int rows;

printf("Enter the number of rows: ");

scanf("%d", &rows);

if (rows <= 0) {

printf("Invalid input! Enter a positive number.\n");

return 1;

}

printPattern(rows);

return 0;

}

Expected output:

\*

\* \*

\* \* \*

#include <stdio.h>

int isPrime(int n) {

if (n <= 1)

return 0;

for (int i = 2; i <= n / 2; i++)

if (n % i == 0)

return 0;

return 1;

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

if (isPrime(num))

printf("%d is a prime number.\n", num);

else

printf("%d is not a prime number.\n", num);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

int main() {

int \*arr;

int size, i;

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

scanf("%d", &size);

arr = (int \*)malloc(size);

if (arr == NULL) {

printf("Memory allocation failed!\n");

return 1;

}

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

arr[i] = i \* 2;

printArray(arr, size); // Undefined function 'printArray'

free(arr);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

int main() {

int r1, c1, r2, c2, i, j, k;

int \*\*m1, \*\*m2, \*\*res;

scanf("%d %d", &r1, &c1);

m1 = malloc(r1 \* sizeof(int \*));

for (i = 0; i < r1; i++) m1[i] = malloc(c1 \* sizeof(int));

for (i = 0; i < r1; i++) for (j = 0; j < c1; j++) scanf("%d", &m1[i][j]);

scanf("%d %d", &r2, &c2);

if (c1 != r2) { printf("Error\n"); return 1; }

m2 = malloc(r2 \* sizeof(int \*));

for (i = 0; i < r2; i++) m2[i] = malloc(c2 \* sizeof(int));

for (i = 0; i < r2; i++) for (j = 0; j < c2; j++) scanf("%d", &m2[i][j]);

res = malloc(r1 \* sizeof(int \*));

for (i = 0; i < r1; i++) res[i] = malloc(c2 \* sizeof(int));

for (i = 0; i < r1; i++) for (j = 0; j < c2; j++) for (k = 0; k < c1; k++) res[i][j] += m1[i][k] \* m2[k][j];

for (i = 0; i < r1; i++) { free(m1[i]); free(res[i]); }

free(m1); free(res);

for (i = 0; i < r2; i++) free(m2[i]);

free(m2);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

int binarySearch(int a[], int l, int r, int t) {

while (l <= r) {

int m = l + (r - l) / 2;

if (a[m] == t) return m;

else if (a[m] < t) l = m + 1;

else r = m - 1;

}

return -1;

}

int main() {

int n, t, i, idx;

int \*arr;

scanf("%d", &n);

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

for (i = 0; i < n; i++) scanf("%d", &arr[i]);

scanf("%d", &t);

idx = binarySearch(arr, 0, n - 1, t);

printf("%d\n", idx);

free(arr);

return 0;

}

#include <stdio.h>

void hanoi(int n, char f, char t, char a) {

if (n == 1) { printf("Move 1 from %c to %c\n", f, t); return; }

hanoi(n - 1, f, a, t);

printf("Move %d from %c to %c\n", n, f, t);

hanoi(n - 1, a, t, f);

}

int main() {

int n;

scanf("%d", &n);

if (n <= 0) { printf("Invalid\n"); return 1; }

printf("Steps:\n");

hanoi(n, 'A', 'C', 'B');

return 0;

}

#include <stdio.h>

#include <stdlib.h>

#define MAX 100

typedef struct { int data[MAX]; int top; } Stack;

void init(Stack \*s) { s->top = -1; }

int empty(Stack \*s) { return s->top == -1; }

int full(Stack \*s) { return s->top == MAX - 1; }

void push(Stack \*s, int v) { if (full(s)) printf("Full\n"); else s->data[++s->top] = v; }

int pop(Stack \*s) { return empty(s) ? -1 : s->data[s->top--]; }

int main() {

Stack s; init(&s);

push(&s, 10); push(&s, 20);

printf("%d\n", pop(&s)); printf("%d\n", pop(&s)); printf("%d\n", pop(&s));

return 0;

}

def calculate\_average(numbers):

**if** not numbers:

**return** None

    total = 0

**for** number **in** numbers

        total += number

    average = total / len(numbers)

**return** average

def main():

     user\_input = input("Enter a list of numbers separated by commas: ")

     numbers = user\_input.split(',')

     numbers = [int(num) **for** num **in** numbers]

     result = calculate\_average(numbers)

**if** result is not None:

         print(f"The average is: {result}")

**else**:

         print("No numbers provided.")

**if** \_\_name\_\_ == "\_\_main\_\_":

#include <stdio.h>

float calculate\_average(int numbers[], int size) {

    if (size == 0) {

        return -1; // Indicate error

    }

    int total = 0;

    for (int i = 0; i <= size; i++) {

        total += numbers[i];

     }

     float average = total / size;

     return average;

 }

 int main() {

     int numbers[] = {1, 2, 3, 4, 5};

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

     float result = calculate\_average(numbers, size);

     if (result != -1) {

         printf("The average is: %.2f\n", result);

     } else {

         printf("No numbers provided.\n");

     }

     return 0;

 }

#include <stdio.h>

void perform\_operations(int a, int b) {

    int sum = a + b;

    int difference = a - b;

    int product = a \* b;

    int quotient = a / b;

    printf("Sum: %d\n", sum);

    printf("Difference: %d\n", difference);

     printf("Product: %d\n", product);

     printf("Quotient: %d\n", quotient);

 }

 int main() {

     int num1, num2;

     printf("Enter two numbers: ");

     scanf("%d %d", &num1, num2);

     perform\_operations(num1, num2);

     return 0;

 }

Expected Output:

Sum: 10

Difference: 10

Product: 0

Quotient: Division by zero is not allowed.