**1. You are given a large integer represented as an integer array digits, where each digits[i] is the ith digit of the integer. The digits are ordered from most significant to least significant in left-to-right order. The large integer does not contain any leading 0's.**

Increment the large integer by one and return the resulting array of digits.

**CODE:**

#include <stdio.h>

int main() {

// Get the number of digits from the user

int size;

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

scanf("%d", &size);

int digits[size];

printf("Enter the digits: ");

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

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

}

int carry = 1;

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

int sum = digits[i] + carry;

digits[i] = sum % 10;

carry = sum / 10;

}

if (carry > 0) {

printf("The result will have no one more digit.\n");

int result[size + 1];

result[0] = carry;

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

result[i + 1] = digits[i];

}

printf("After incrementing: ");

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

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

}

printf("\n");

} else {

printf("After incrementing by one: ");

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

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

}

printf("\n");

}

return 0;

}

**2. You are given an integer array nums. You are initially positioned at the array's first index, and each element in the array represents your maximum jump length at that position. Return true if you can reach the last index, or false otherwise.**

**CODE:**

#include <stdbool.h>

#include <stdio.h>

bool canJump(int\* nums, int numsSize) {

int maxReach = 0;

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

if (i > maxReach) {

return false;

}

maxReach = fmax(maxReach, i + nums[i]);

if (maxReach >= numsSize - 1) {

return true;

}

}

return false;

}

int main() {

int size;

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

scanf("%d", &size);

int nums[size];

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

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

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

}

bool result = canJump(nums, size);

printf("The result is: %s\n", result ? "true" : "false");

return 0;

}

**3.  Given an integer array nums, find the subarray with the largest sum, and return its sum.**

**CODE:**

#include <stdio.h>

#include <limits.h>

int maxSubArray(int\* nums, int numsSize) {

int maxSum = INT\_MIN;

int currentSum = 0;

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

currentSum = fmax(nums[i], currentSum + nums[i]);

maxSum = fmax(maxSum, currentSum);

}

return maxSum;

}

int main() {

int size;

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

scanf("%d", &size);

int nums[size];

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

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

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

}

int result = maxSubArray(nums, size);

printf("The maximum subarray sum is: %d\n", result);

return 0;

}