1 ) Dinamik Bellek Kullanarak Dizi Kopyalama : Kullanıcının girdiği bir tamsayı dizisini **dinamik bellek kullanarak** kopyalayan bir fonksiyon yazın. Bu fonksiyon, **orijinal diziyi değiştirmemeli** ve yeni kopyalanmış dizinin adresini döndürmelidir.

**Cevap:**

#include <stdio.h>

#include <stdlib.h>

// Dizi kopyalayan fonksiyon (pointer kullanıyor)

int\* copyArray(int \*arr, int size) {

 int \*newArr = (int \*)malloc(size \* sizeof(int)); // Dinamik bellek ayırma

 if (newArr == NULL) {

 printf("Bellek tahsis hatası!\n");

 return NULL;

 }

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

 \*(newArr + i) = \*(arr + i); // Elemanları kopyala

 }

 return newArr; // Yeni dizinin adresini döndür

}

// Diziyi ekrana yazdıran fonksiyon

void printArray(int \*arr, int size) {

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

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

 }

 printf("\n");

}

int main() {

 int numbers[] = {10, 20, 30, 40, 50};

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

 printf("Orijinal dizi: ");

 printArray(numbers, size);

 int \*copiedArray = copyArray(numbers, size); // Yeni dizi kopyalandı

 if (copiedArray != NULL) {

 printf("Kopyalanmış dizi: ");

 printArray(copiedArray, size);

 free(copiedArray); // Belleği serbest bırak

 }

 return 0;

}

2 ) Karakter Dizisinde Geçen En Uzun Kelimeyi Bulma : Kullanıcıdan alınan bir metinde **pointer kullanarak en uzun kelimeyi bulan bir fonksiyon** yazın.

**Cevap:**

#include <stdio.h>

#include <string.h>

// En uzun kelimeyi bulan fonksiyon

void findLongestWord(char \*str, char \*\*longestWord, int \*maxLength) {

 char \*current = str, \*wordStart = NULL; // 'current' stringi taramak için, 'wordStart' kelime başlangıcını tutar

 int length = 0, maxLen = 0; // 'length' mevcut kelimenin uzunluğu, 'maxLen' en uzun kelimenin uzunluğunu tutar

 while (\*current != '\0') { // Stringin sonuna kadar ilerle

 if (\*current != ' ' && \*current != '\n' && \*current != '\t') { // Boşluk, yeni satır veya tab karakteri değilse

 if (wordStart == NULL) {

 wordStart = current; // Eğer yeni bir kelimeye başlanıyorsa başlangıç adresini kaydet

 }

 length++; // Kelimenin uzunluğunu artır

 } else { // Boşluk veya özel karakter geldiğinde kelime sonlanır

 if (wordStart != NULL && length > maxLen) {

 \*longestWord = wordStart; // Eğer mevcut kelime önceki en uzundan daha uzunsa güncelle

 \*maxLength = length;

 maxLen = length;

 }

 wordStart = NULL; // Yeni kelimeyi başlatmak için sıfırla

 length = 0; // Uzunluğu sıfırla

 }

 current++; // Bir sonraki karaktere geç

 }

 // Son kelimenin kontrolü (Eğer en uzun kelime metnin sonundaysa)

 if (wordStart != NULL && length > maxLen) {

 \*longestWord = wordStart;

 \*maxLength = length;

 }

}

int main() {

 char text[] = "Bu bir pointer ile en uzun kelimeyi bulan programdır"; // Örnek metin

 char \*longestWord; // En uzun kelimenin başlangıç adresini tutacak pointer

 int maxLength = 0; // En uzun kelimenin uzunluğunu tutacak değişken

 findLongestWord(text, &longestWord, &maxLength); // En uzun kelimeyi bul

 printf("En uzun kelime: ");

 for (int i = 0; i < maxLength; i++) { // Bulunan kelimeyi yazdır

 printf("%c", longestWord[i]);

 }

 printf("\nUzunluk: %d\n", maxLength); // En uzun kelimenin uzunluğunu ekrana yazdır

 return 0; // Programı başarıyla sonlandır

}