

ISPARTA UYGULAMALI BİLİMLER ÜNİVERSİTESİ - TEKNOLOJİ FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ
BÖLÜMÜ

2024-2025 BAHAR DÖNEMİ BLG-102 VERİ YAPILARI DERSİ ÖDEV RAPORU

Ad Soyad: Ceren Mıtırık

Numara: 2212721032

(Bu bölüme ödev numarasını ve ödevi yazın.)

ÖDEV :

ÖRNEK 7:

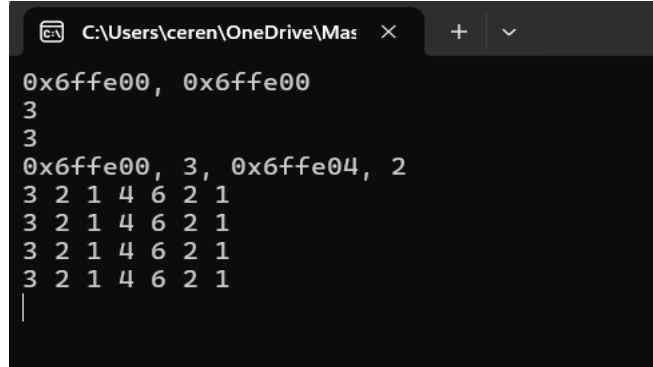
Kodlar 7.1:

```
int *buyuk(int *x, int *y) {
    if (*x > *y) // x pointer'ının işaret ettiği değer, y pointer'ının işaret ettiği değerden büyükse
        return x; // x pointer'ı döndürülüyor yani büyük olan sayının adresi
    else
        return y; // eğer büyük değilse y pointer'ı döndürülüyor
}

int main() {
    int a, b;
    int *p; // int pointer'ı tanımla
    cout << "iki deger giriniz: ";
    cin >> a >> b; // Kullanıcının girdiği iki değer a ve b değişkenlerine atanıyor.
    p = buyuk(&a, &b); // fonksiyon çağrılıyor ve a ve b değişkenlerinin adresleri parametre olarak veriliyor ve döndürülen adres p pointer'ına atanıyor.
    cout << "girilen degerlerden buyuk olan: " << *p;

    getch();
    return 0;
}
```

Çıktı 7.2:



```
C:\Users\ceren\OneDrive\Mas x + v
0x6ffe00, 0x6ffe00
3
3
0x6ffe00, 3, 0x6ffe04, 2
3 2 1 4 6 2 1
3 2 1 4 6 2 1
3 2 1 4 6 2 1
3 2 1 4 6 2 1
3 2 1 4 6 2 1
|
```

ÖRNEK 8:

Kodlar 8.1:

```
int main() {
    // pointer kullanarak string boyutunu bulma

    char a[] = "bu bir string.";
    char *p; // char pointer'ı tanımlanıyor.
    int n = 0; // string boyutu sayacı=0
    p = a; // p pointer'ına a dizisinin başlangıç adresi atanıyor.
    while (*p++ != '\0') // p pointer'ının işaret ettiği karakter null karakteri olana kadar döngü devam etsin
        n++; // String boyutu sayacı bir artır
    cout << "Strigin boyutu: " << n << "\n";

    while (*p++) // p pointer'ı null karakteri geçtikten sonra da döngü devam ediyor ?
        n++; // String boyutu sayacı tekrar artır
    cout << "Strigin boyutu: " << n << "\n";

    getch();
    return 0;
}
```

Çıktılar 8.2:

```
C:\Users\ceren\OneDrive\Mas X + v
Strigin boyutu: 14
Strigin boyutu: 14
```

ÖRNEK 9:

Kodlar 9.1:

```
int en_buyuk(int x, int boyut, int *max) { // fonksiyon tanımla int pointer'ı alsın ve int pointer'ı döndürsün
    int *p;
    *max = *x; // max pointer'ının işaret ettiği değere, x pointer'ının işaret ettiği ilk değeri ata
    for (p = x; p < x + boyut; p++) // x pointer'ından başlayarak, boyut kadar ilerleyen bir döngü oluştur
        if (*p > *max) // p pointer'ının işaret ettiği değer, max pointer'ının işaret ettiği değerden büyükse alt satıra geç
            *max = *p; // p pointer'ının işaret ettiği değeri max pointer'ının işaret ettiği değere ata
    return max; // en büyük değerin adresini döndür
}

int main() {
    int a[] = { 2, 4, 1, 6, 5, 8, 3, 2, 5, 6 };
    int enbuyuk_deger;
    int *p; // int pointer'ı tanımla
    p = en_buyuk(a, sizeof(a) / sizeof(a[0]), &enbuyuk_deger); // fonksiyonu çağır ve a dizisi, dizinin boyutu ve enbuyuk_deger değişkeninin adresi parametre olarak ver

    cout << "dizinin en büyük elemanı:" << *p << "\n"; // p pointer'ının işaret ettiği değeri yazdır
    cout << "dizinin en büyük elemanı:" << enbuyuk_deger << "\n";
    getch();
    return 0;
}
```

Çıktılar 9.2

```
C:\Users\ceren\OneDrive\Mas X + v
dizinin en büyük elemanı:8
dizinin en büyük elemanı:8
```

ÖRNEK 10:

Kodlar 10.1:

```
char birlestir(char *x, char *y, char *z) { // char pointer alan ve char pointer döndüren fonksiyon tanımla
    char *p;
    for(p = x; *p != '\0'; p++, z++) // x pointer'ının işaret ettiği string'in sonuna kadar döngü
        *z = *p; // x pointer'ının işaret ettiği karakter, z pointer'ının işaret ettiği yere ata
    for(p = y; *p != '\0'; p++, z++) // y pointer'ının işaret ettiği string'in sonuna kadar döngü
        *z = *p; // y pointer'ının işaret ettiği karakter, z pointer'ının işaret ettiği yere ata
    *z = '\0'; // z pointer'ının işaret ettiği string'in sonuna null karakteri ekle
}

int main() {
    char str1[] = "abc", str2[] = "xyz"; // char dizileri tanımla
    char birlesme[10]; // 10 elemanlı char dizisi tanımla
    birlestir(str1, str2, birlesme); // fonksiyonu çağır ve dizileri parametre olarak ver
    cout << "birlestirilen string: " << birlesme << "\n";

    getch();
    return 0;
}
```

Çıktılar 10.2:

```
C:\Users\ceren\OneDrive\Mas x + v
birlestirilen string: abcxyz
```

ÖRNEK 11:

Kodlar 11.1:

```
char *birlestir(char *x, char *y, char *z) { // char pointer alan ve char pointer donduren birlestir fonksiyonu tanımla
    char *p;
    char *q = z; // z pointer'ının adresini q pointer'ına ata
    for(p = x; *p != '\0'; p++, z++) // x pointer'ının işaret ettiği string'in sonuna kadar döngü
        *z = *p; // x pointer'ının işaret ettiği karakter, z pointer'ının işaret ettiği yere ata

    for(p = y; *p != '\0'; p++, z++) // y pointer'ının işaret ettiği string'in sonuna kadar döngü
        *z = *p; // y pointer'ının işaret ettiği karakter, z pointer'ının işaret ettiği yere ata
    *z = '\0'; // z pointer'ının işaret ettiği string'in sonuna null karakteri ekle
    return q; // q pointer'ı döndür
}

int main() {
    char str1[] = "abc", str2[] = "xyz";
    char birlesme[10];
    char *q;
    q = birlestir(str1, str2, birlesme); // fonksiyon çağır ve str1, str2 ve birlesme dizileri parametre olarak ver, döndürülen değer q'ya ata
    cout << "birlestirilen string: " << q << "\n"; // q pointer'ının işaret ettiği stringi yazdır

    getch();
    return 0;
}
```

Çıktılar 11.2:

```
C:\Users\ceren\OneDrive\Mas x + v
birlestirilen string: abcxyz
```

ÖRNEK 12:

Kodlar 12.1:

```
int main() {
    char *isimler[] = { "Mehmet", "Ahmet", "Ali", "Ayse" };
    int i; // sayaç

    cout << sizeof(i) << "\n"; // int tipinin bellekte kapladığı boyutu yaz
    cout << sizeof(char*) << "\n"; // char pointer'ın bellekte kapladığı boyutu yaz
    cout << sizeof(isimler) << "\n"; // isimler dizisinin bellekte kapladığı toplam boyutu yaz

    cout << "Mehmet ifadesinin bellekteki baslangic adresi: " << (void*)isimler[0] << "\n"; // isimler[0] tuttuğu adres yaz
    cout << "Ahmet ifadesinin bellekteki baslangic adresi: " << (void*)isimler[1] << "\n"; // isimler[1]'in tuttuğu adres yaz
    cout << "Ali ifadesinin bellekteki baslangic adresi: " << (void*)isimler[2] << "\n"; // isimler[2]'nin tuttuğu adres yaz
    cout << "Ayse ifadesinin bellekteki baslangic adresi: " << (void*)isimler[3] << "\n"; // isimler[3]'ün tuttuğu adres yaz

    cout << isimler[1] - isimler[0] << endl; // Ahmet ve Mehmet'in başlangıç adresleri arasındaki fark byte cinsinden yaz
    cout << isimler[2] - isimler[1] << endl; // Ali ve Ahmet'in başlangıç adresleri arasındaki fark byte cinsinden yaz
    cout << isimler[3] - isimler[2] << endl; // Ayse ve Ali'nin başlangıç adresleri arasındaki fark byte cinsinden yaz

    // Diziler birbirine doğrudan atanamaz, diziye işaret eden pointer'lar aracılığıyla atama yapılabilir.
    isimler[2] = isimler[3]; // Ali'nin yerine Ayse'nin adresini ata

    for (i = 0; i < 4; i++) {
        cout << isimler[i] << ", ";
    }

    return 0;
}
```

Çıktılar 12.2:

```
C:\Users\ceren\OneDrive\Mas  X  +  v
4
8
32
Mehmet ifadesinin bellekteki baslangic adresi: 0x488000
Ahmet ifadesinin bellekteki baslangic adresi: 0x488007
Ali ifadesinin bellekteki baslangic adresi: 0x48800d
Ayse ifadesinin bellekteki baslangic adresi: 0x488011
7
6
4
Mehmet, Ahmet, Ayse, Ayse,
```

(Bu bölüme sıradaki ödev numarasını ve ödevi yazın.)

ÖDEV :

--