2024-2025 BAHAR DÖNEMİ

ALGORİTMALAR VE PROGRAMLAMA II

UYGULAMA 7

1. Soru:
Bir dosyadan (**musteri.dat**) müşteri kayıtlarını okuyan ve yöneten bir C programı yazınız.
Her kayıt, aşağıdaki bilgileri içermelidir:

- Hesap numarası (**int**),
- Müşteri adı (**string**),
- Bakiye (**double**).

Örnek Dosya İçeriği

```
1001 Ali 500.0
1002 Veli -250.0
1003 Ayşe 0.0
1004 Zeynep 1200.5
1005 Mehmet -50.5
```

Program kullanıcıya aşağıdaki işlemleri sunmalıdır:

- Bakiye değeri sıfır (0) olan hesapları listele,
- Negatif bakiyeye sahip (borçlu) hesapları listele,
- Pozitif bakiyeye sahip hesapları listele,
- Yeni müşteri kaydı ekle,
- Var olan bir müşteri kaydını sil,
- Programı sonlandır.

Beklenen ekran çıktısı:

```
Bir seçim yapınız:

1 - Bakiyesi sıfır olan hesapları listele

2 - Negatif bakiyeye sahip hesapları listele

3 - Pozitif bakiyeye sahip hesapları listele

4 - Yeni müşteri kaydı ekle

5 - Müşteri kaydı sil

6 - Çıkış

Seçiminiz: 1


Bakiyesi sıfır olan hesaplar:

Hesap No: 1003 İsim: Ayşe Bakiye: 0.00
```

İpucu:

- `struct` yapısını kullanarak müşteri kayıtlarını temsil ediniz.
- `fred()` fonksiyonu ile dosyadan veri okuyunuz.
- Yeni müşteri eklerken `fwrite()` kullanarak dosyaya kayıt yazınız.
- Müşteri kaydı silerken ilgili kayıt bilgilerini sıfırlayarak dosyayı güncelleyiniz.
- `rewind()` fonksiyonunu kullanarak her işlem öncesi dosya başına dönünüz.
- Menü seçeneklerini `switch-case` yapısı kullanarak yönetiniz.

Not:
Yeni müşteri kaydı eklerken kullanıcıdan hesap numarası, isim ve bakiye bilgisi alınmalı; Müşteri silme işleminde silinecek hesap numarası istenmeli ve kayıt dosyada "silindi" olarak işaretlenmelidir.

ÇÖZÜM:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Müşteri kaydını temsil eden yapı (struct)
struct musteri {
    int hesapNo;
    char ad[30];
    double bakiye;
};

// Fonksiyon prototipleri
void bakiyeSifirListele(FILE *dosya);
void negatifBakiyeListele(FILE *dosya);
void pozitifBakiyeListele(FILE *dosya);
void yeniMusteriEkle(FILE *dosya);
void musteriSil(FILE *dosya);

int main() {
    FILE *mfPtr;
    int secim;

    // Dosyayı binary modda hem okuyup hem yazmak için açıyoruz
    if ((mfPtr = fopen("musteri.dat", "rb+")) == NULL) {
        printf("Dosya açılamadı.\n");
        return 1;
    }

    do {
        printf("\nBir seçim yapınız:\n");
        printf("1 - Bakiyesi sıfır olan hesapları listele\n");
        printf("2 - Negatif bakiyeye sahip hesapları listele\n");
        printf("3 - Pozitif bakiyeye sahip hesapları listele\n");
        printf("4 - Yeni müşteri kaydı ekle\n");
        printf("5 - Müşteri kaydı sil\n");
        printf("6 - Programı sonlandır\n");
        printf("Seçiminiz: ");
        scanf("%d", &secim);

        switch (secim) {
            case 1:
                bakiyeSifirListele(mfPtr);
                break;
            case 2:
                negatifBakiyeListele(mfPtr);
                break;
            case 3:
                pozitifBakiyeListele(mfPtr);
                break;
            case 4:
                yeniMusteriEkle(mfPtr);
                break;
            case 5:
                musteriSil(mfPtr);
                break;
            case 6:
                printf("\nProgram sonlandırılıyor...\n");
                break;
            default:
                printf("\nGeçersiz seçim. Lütfen 1-6 arasında bir değer giriniz.\n");
                break;
        }
    } while (secim != 6);

    fclose(mfPtr);
    return 0;
}

// Bakiyesi sıfır olan hesapları listele
void bakiyeSifirListele(FILE *dosya) {
    struct musteri m;
    rewind(dosya);
    printf("\nBakiyesi sıfır olan hesaplar:\n");

    while (fread(&m, sizeof(struct musteri), 1, dosya) == 1) {
        if (m.bakiye == 0.0 && m.hesapNo != 0) {
            printf("Hesap No: %d İsim: %s Bakiye: %.2lf\n", m.hesapNo, m.ad, m.bakiye);
```

```
        }
    }
}

// Negatif bakiyeye sahip hesapları listele
void negatifBakiyeListele(FILE *dosya) {
    struct musteri m;
    rewind(dosya);
    printf("\nNegatif bakiyeye sahip hesaplar:\n");

    while (fread(&m, sizeof(struct musteri), 1, dosya) == 1) {
        if (m.bakiye < 0.0 && m.hesapNo != 0) {
            printf("Hesap No: %d İsim: %s Bakiye: %.2lf\n", m.hesapNo, m.ad, m.bakiye);
        }
    }
}

// Pozitif bakiyeye sahip hesapları listele
void pozitifBakiyeListele(FILE *dosya) {
    struct musteri m;
    rewind(dosya);
    printf("\nPozitif bakiyeye sahip hesaplar:\n");

    while (fread(&m, sizeof(struct musteri), 1, dosya) == 1) {
        if (m.bakiye > 0.0 && m.hesapNo != 0) {
            printf("Hesap No: %d İsim: %s Bakiye: %.2lf\n", m.hesapNo, m.ad, m.bakiye);
        }
    }
}

// Yeni müşteri kaydı ekle
void yeniMusteriEkle(FILE *dosya) {
    struct musteri yeni;

    printf("\nYeni müşteri bilgilerini giriniz:\n");
    printf("Hesap No: ");
    scanf("%d", &yeni.hesapNo);
    printf("İsim: ");
    scanf("%s", yeni.ad);
    printf("Bakiye: ");
    scanf("%lf", &yeni.bakiye);

    fseek(dosya, 0, SEEK_END);
    fwrite(&yeni, sizeof(struct musteri), 1, dosya);

    printf("\nYeni müşteri kaydı başarıyla eklendi.\n");
}

// Müşteri kaydı sil (hesapNo ve bakiye 0 yapılır)
void musteriSil(FILE *dosya) {
    struct musteri m;
    int silinecekNo;
    long pos;

    printf("\nSilinecek müşteri hesap numarasını giriniz: ");
    scanf("%d", &silinecekNo);

    rewind(dosya);
    while (fread(&m, sizeof(struct musteri), 1, dosya) == 1) {
        if (m.hesapNo == silinecekNo) {
            pos = ftell(dosya) - sizeof(struct musteri); // bulunduğumuz kaydın başlangıcı
            m.hesapNo = 0;
            strcpy(m.ad, "silindi");
            m.bakiye = 0.0;

            fseek(dosya, pos, SEEK_SET);
            fwrite(&m, sizeof(struct musteri), 1, dosya);
            printf("\nMüşteri kaydı başarıyla silindi.\n");
            return;
        }
    }

    printf("\nBelirtilen hesap numarası bulunamadı.\n");
}
```