buffer.c File Reference

Dairesel (circular) karakter tamponu (FIFO) uygulamasi. More...

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

Go to the source code of this file.

Macros

`#define BUYUKLUK 10`
Tamponun (buffer) maksimum kapasitesi.

Functions

`int push (char value)`
Buffer'a karakter ekler (push islemi)
`int pop (char *value)`
Buffer'dan karakter okur (pop islemi)
`void buffer_goster ()`
Buffer'in mevcut durumunu grafiksel olarak ekrana yazdirir.
`int main ()`
Programin giris noktasi (main fonksiyonu)

Variables

`char buffer [BUYUKLUK]`
Dairesel tamponu temsil eden global degiskenler.
`int head = 0`
`int tail = 0`
`int count = 0`

Detailed Description

Dairesel (circular) karakter tamponu (FIFO) uygulamasi.

Bu program, sabit boyutlu bir tampon (buffer) kullanarak karakter ekleme ve cikarma islemleri yapar. Tampon bir kuyruk (queue) mantigi ile calisir: Ilk giren karakter ilk cikar (FIFO).

Kullaniciya bir menu sunulur. Bu menu araciligiyla kullanici:

- Coklu karakter girisi yaparak buffer'a veri ekleyebilir,

- Mevcut buffer'dan bir karakter okuyabilir (cikarabilir),
- Tamponun mevcut durumunu gorebilir,
- Veya cikis yapabilir.

Definition in file **buffer.c**.

Macro Definition Documentation

◆ BUYUKLUK

```
#define BUYUKLUK   10
```

Tamponun (buffer) maksimum kapasitesi.

< Giris/Cikis islemleri icin gerekli kutuphane < Karakter dizisi islemleri icin kutuphane

Definition at line **22** of file **buffer.c**.

Function Documentation

◆ buffer_goster()

`void buffer_goster ( )`

Buffer'in mevcut durumunu grafiksel olarak ekrana yazdirir.

Dolular karakterlerle, bos alanlar '-' ile gosterilir.

Definition at line **73** of file **buffer.c**.

```
73                      {
74      printf("Buffer durumu: [");
75      for (int i = 0; i < BUYUKLUK; i++) {
76          if (count == 0) {
77              printf(" -");
78          } else {
79              int aktif = 0;
80              if (tail < head) {
81                  if (i >= tail && i < head) aktif = 1;
82              } else if (tail > head) {
83                  if (i >= tail || i < head) aktif = 1;
84              } else if (count == BUYUKLUK) {
85                  aktif = 1;
86              }
87
88              if (aktif) {
89                  printf(" %c", buffer[i]);
90              } else {
91                  printf(" -");
92              }
93          }
94      }
95      printf(" ]\n");
96  }
```

◆ `main()`

```
int main ( )
```

Programin giris noktasi (main fonksiyonu)

Kullaniciya etkilesimli bir menu sunar.

1. Coklu karakter ekleme
2. FIFO'dan karakter okuma

1. Programdan cikis

Returns

int Programdan cikis kodu

Definition at line **108** of file **buffer.c**.

```
108             {
109     int secim;
110     char giris[100];
111     char alinan;
112 
113     while (1) {
114         printf("\n1 - Ekle (coklu harf veya rakam)\n2 - Oku (cikart)\n0 - Cikis\nS
    ecim: ");
115         scanf("%d", &secim);
116         while ((getchar()) != '\n'); // Enter karakterini temizle
117 
118         switch (secim) {
119             case 1:
120                 printf("Giris (birden fazla harf/rakam yaz): ");
121                 fgets(giris, sizeof(giris), stdin);
122                 giris[strcspn(giris, "\n")] = '\0'; // Satir sonunu kaldir
123 
124                 for (int i = 0; i < strlen(giris); i++) {
125                     if (push(giris[i])) {
126                         printf("Eklendi: %c\n", giris[i]);
127                     } else {
128                         printf("Buffer dolu, eklenemedi: %c\n", giris[i]);
129                         break;
130                     }
131                 }
132                 buffer_goster();
133                 break;
134 
135             case 2:
136                 if (pop(&alinan)) {
137                     printf("Okunan karakter: %c\n", alinan);
138                 } else {
139                     printf("Buffer bos!\n");
140                 }
141                 buffer_goster();
142                 break;
143 
144             case 0:
145                 printf("Programdan cikiliyor...\n");
146                 return 0;
147 
148             default:
149                 printf("Gecersiz secim!\n");
150         }
151     }
```

```
152
153     return 0;
154 }
```

◆ pop()

```
int pop ( char * value )
```

Buffer'dan karakter okur (pop islemi)

Eger buffer bossa karakter okunamaz.

Parameters

value Okunan karakterin yazilacagi adres

Returns

int 1 basari, 0 basarisiz (buffer bos)

Definition at line **58** of file **buffer.c**.

```
58                      {
59     if (count == 0) {
60         return 0; // Bos
61     }
62     *value = buffer[tail];
63     tail = (tail + 1) % BUYUKLUK;
64     count--;
65     return 1;
66 }
```

◆ push()

```
int push ( char value )
```

Buffer'a karakter ekler (push islemi)

Eger buffer doluysa karakter eklenemez.

Parameters

value Eklenecek karakter

Returns

int 1 basari, 0 basarisiz (buffer dolu)

Definition at line **40** of file **buffer.c**.

```
40                      {
41      if (count == BUYUKLUK) {
42          return 0; // Dolu
43      }
44      buffer[head] = value;
45      head = (head + 1) % BUYUKLUK;
46      count++;
47      return 1;
48  }
```

Variable Documentation

◆ buffer

```
char buffer[BUYUKLUK]
```

Dairesel tamponu temsil eden global degiskenler.

Karakterleri tutacak olan buffer

Definition at line **27** of file **buffer.c**.

◆ count

```
int count = 0
```

Tampondaki mevcut karakter sayisi

Definition at line **30** of file **buffer.c**.

◆ head

```
int head = 0
```

Yeni karakterin eklenecegi konum

Definition at line **28** of file **buffer.c**.

◆ tail

```
int tail = 0
```

Okunacak karakterin konumu

Definition at line **29** of file **buffer.c**.