Karadeniz Teknik Üniversitesi
Bilgisayar Mühendisliği Bölümü
Öğr.Gör. Ömer ÇAKIR

BIL 2001 Veri Yapıları
Arasınav, 16.11.2017, 13:00
Süre : **90** Dakika

CEVAPLAR

```
void SinglyLinkedList::removeBack()
{
   if (head == NULL)
   {
      cout << "List is empty !" << endl;
      return;
   }

   SinglyNode* prev = head;

   if (prev->next == NULL)
   {
      head = NULL;
      delete prev;
   }
   else
   {
      while (.....................)

            ......................

      ......................

      ......................

   }
}
```

1. **removeBack()** fonksiyonundaki ..... satırlarına aşağıdaki kodlardan hangisi yazılmalıdır? **(25P)**

*Yanlış cevaptan **5P** kırılacaktır.*

```
(A)  while (prev->next->next != NULL)
           prev = prev->next;
     prev->next = NULL;
     delete prev->next;

(B)  while (prev->next->next != NULL)
           prev = prev->next;
     delete prev->next;
     prev->next = NULL;

(C)  while (prev->next != NULL)
           prev = prev->next;
     delete prev->next;
     prev->next = NULL;

(D)  while (prev->next != NULL)
           prev = prev->next;
     prev->next = NULL;
     delete prev->next;
```

2. İkili ağaçtan "çocuklu" düğüm silinmelerinde dengeli bir ağaç elde etmek üzere düğüm silme algoritmasında bir iyileştirme yöntemi öneriniz. **(25P)**

1. İyileştirme
Bilindiği gibi ikili ağaçtan çocuklu düğümler silinirken yerine ya kendisinden büyük en küçük (KBEK) ya da kendisinden küçük en büyük (KKEB) düğüm yazılır.
Ağaçtan silinecek düğümün sol alt ağacındaki ve sağ alt ağacındaki toplam düğüm sayıları karşılaştırılır. Sol alt ağaçtaki düğüm sayısı fazla ise KKEB; sağ alt ağaçtaki düğüm sayısı fazla ise de KBEK uygulanır. Alt ağaçlardaki düğüm sayıları, ağaçta gezinme fonksiyonlarından herhangi birinde küçük bir değişiklik yapılarak hesaplanabilir.

2. İyileştirme
Düğüm (node) structure'ına sol ve sağ alt ağaçlardaki düğümlerin sayısını tutacak birer int değişken eklenir ve düğümler ağaca eklenirken değişkenler ++ yapılarak güncellenir. Herhangi bir düğüm silinirken bu değişkenler karşılaştırılır ve KKEB veya KBEK yaklaşımlarından hangisinin uygulanacağına karar verilir.

```
void tree(int i, int j, int p, int n, int k)
{
      if (i < 1) return;

      ..................

      if (n == k)
            ..............................
      else
            tree(i, j + p, p, n + 1, k);
}

int main()
{
      tree(8, 8, 16, 1, 1);
}
```

3. Programda ..... ile temsil edilen satırlar aşağıdakilerden hangisi olursa çıktı **8 4 12 2 6 10 14 1 3 5 7 9 11 13 15** şeklinde olur? **(25P)**

*Yanlış cevaptan **5P** kırılacaktır.*

(A)
```
cout << i << endl;
tree(i / 2, i / 2, p / 2, 1, k * 2);
```

(B)
```
cout << i << endl;
tree(i / 2, j / 2, p / 2, 1, k * 2);
```

(C)
```
cout << j << endl;
tree(i / 2, i / 2, p / 2, 1, k * 2);
```

(D)
```
cout << j << endl;
tree(i / 2, j / 2, p / 2, 1, k * 2);
```

```
void print(DoublyNode* first, DoublyNode* last)
{
   if ((first->elem.compare(last->elem)== 0)
      && (first->score == last->score))
         cout << first->elem << endl;
   else
      print(first->next, last->prev);
}

int main()
{
   DoublyLinkedList list;

   list.insertOrdered("Paul", 720);
   list.insertOrdered("Rose", 590);
   list.insertOrdered("Anna", 660);
   list.insertOrdered("Mike", 1105);
   list.insertOrdered("Rob",  750);
   list.insertOrdered("Jack", 510);
   list.insertOrdered("Jill", 740);

   list.print(
     list.header->next,
     list.trailer->prev);
}
```

4. Yukarıdaki programın çıktısı nedir? **(25P)**

Paul