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

BIL 1052 Veri Yapıları
Bütünleme Sınavı, 22.01.2016, 10:00, D2-3
Süre : **90** Dakika

CEVAPLAR

```
void reverseList(DoublyLinkedList* list,
                 DoublyNode* hNext,
                 DoublyNode* tPrev)
{
  if (hNext == tPrev) return;

  if (hNext->next == tPrev)
  {
    list->add(hNext, tPrev->elem, tPrev->score);
    list->remove(tPrev);
    return;
  }
  else
  {
    list->add(hNext, tPrev->elem, tPrev->score);
    tPrev = tPrev->prev;
    list->remove(tPrev->next);

    list->add( ....., hNext->elem, hNext->score);
    hNext = hNext->next;
    list->remove(hNext->prev);

    reverseList(list, hNext, ..... );
  }
}

void main()
{
  DoublyLinkedList* list = new DoublyLinkedList();
  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);

  reverseList(list,
              list->header->next,
              list->trailer->prev);
  list->printH2T();
}
```

1. **reverseList()** fonksiyonunda ..... ile temsil edilen satırlar aşağıdaki seçeneklerden hangi ikisi olduğunda **printH2T()** aynı çıktıyı verir? Çıktı nedir? **(40P)**

(A) tPrev
tPrev

(B) tPrev->next
tPrev->next

(C) tPrev->next
tPrev->prev

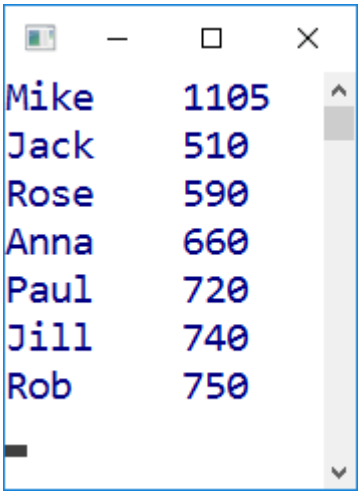
(D) tPrev->next
tPrev

(E) tPrev
tPrev->prev

(F) tPrev->prev
tPrev

Eşdeğer seçenekler → (B) ve (E)

Çıktı :



```
void LinkedBinaryTree::traverse(Node* p)
{
   while (root != NULL)
   {
      while (p->left != NULL) p = p->left;
      cout << p->elt << endl;
      deleteNode(root, p->elt);
      p = root;
   }
}

void main()
{                                          // Çıktı
   LinkedBinaryTree Tree;                  1
   Tree.addRoot();                         2
   Tree.root->elt = 8;                     3
   Tree.addBelowRoot(Tree.root,  4);       4
   Tree.addBelowRoot(Tree.root, 12);       5
   Tree.addBelowRoot(Tree.root,  2);       6
   Tree.addBelowRoot(Tree.root,  6);       7
   Tree.addBelowRoot(Tree.root, 10);       8
   Tree.addBelowRoot(Tree.root, 14);       9
   Tree.addBelowRoot(Tree.root,  1);       10
   Tree.addBelowRoot(Tree.root,  3);       11
   Tree.addBelowRoot(Tree.root,  5);       12
   Tree.addBelowRoot(Tree.root,  7);       13
   Tree.addBelowRoot(Tree.root,  9);       14
   Tree.addBelowRoot(Tree.root, 11);       15
   Tree.addBelowRoot(Tree.root, 13);
   Tree.addBelowRoot(Tree.root, 15);

   binaryTree.traverse(binaryTree.root);
}
```

2. **a)** Yukarıdaki programın çıktısı nedir? **(20P)**

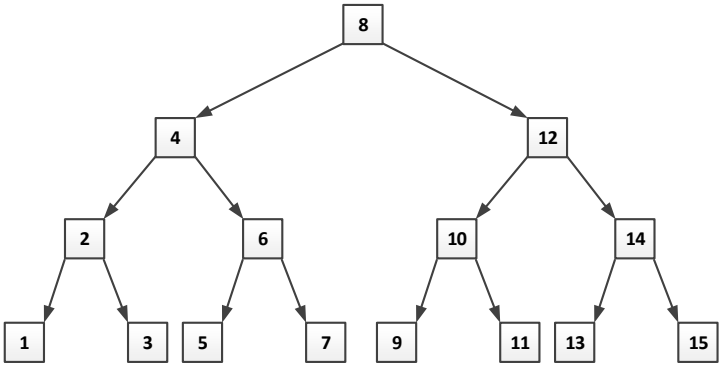
Not→ Silinen düğümün yerine kendisinden büyük en küçük düğümün geldiğini varsayınız.

b) Yukarıdaki programın çıktısı hangi ağaç gezinme yöntemine eşdeğerdir? **(20P)** *Yanlış cevaptan **5P** kırılacaktır.*

(A) inorder

(B) preorder

(C) postorder




```
void insertOrdered(string& e, int& i)
{
   CircularlyNode* newNode = new CircularlyNode;
   newNode->elem           = e;
   newNode->score          = i;

   if (cursor == NULL)
   {
      newNode->next  = newNode;
      cursor         = newNode;
      return;
   }

   CircularlyNode* front   = cursor->next;
   CircularlyNode* back    = cursor;

   while( newNode->score > front->score )
   {
      back = front;
      front = front->next;

      if (.......................) break;
   }

   back->next        = newNode;
   newNode->next     = front;

   if (newNode->score > cursor->score)
       cursor = cursor->next;
}
```

3. Düğümleri dairesel bağlı listeye **score** değerlerine göre küçükten büyüğe sıralı ekleyen **insertOrdered()** fonksiyonunda **.....** ile temsil edilen satır için aşağıda önerilen kodların başına doğru ise **D**; hatalı ise **H** yazınız. **(20P)** *Yanlış cevabın herbirinden **5P** kırılacaktır.*

(D) back == cursor

(H) back == cursor->next

(H) front == cursor

(D) front == cursor->next