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

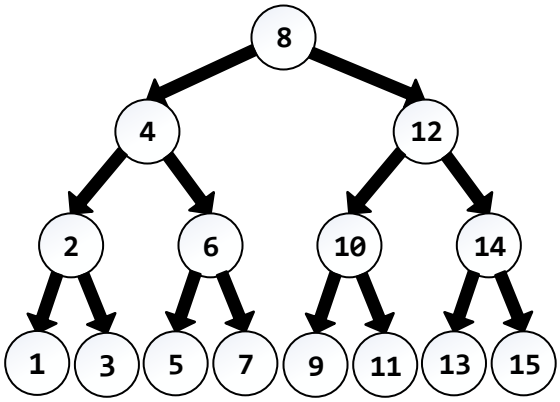
BIL 2001 Veri Yapıları
Final Sınavı, 07.01.2017, 10:00, D-2, D-9
Süre : **120** Dakika

CEVAPLAR

```
void traverse(Node* v)
{
      if (v->left != NULL)
            traverse(v->left);
      else
      if (v->right == NULL)
            cout << v->elt << " ";

      if (v->right != NULL)
            traverse(v->right);
}
```

1. **main()**'de aşağıdaki ağacın **root**u ile çağrıldığı varsayılan **traverse()** fonksiyonunun çıktısı nedir? **(25P)**



1 3 5 7 9 11 13 15

```
void traverse(Node* v)
{
   stack<Node*> stl_stack;

   stl_stack.push(v);

   while (!stl_stack.empty())
   {
       Node* current = stl_stack.top();
       cout << current->elt << " ";

       stl_stack.pop();

       if (current->right != NULL)
             stl_stack.push(current->right);

       if (current->left != NULL)
             stl_stack.push(current->left);
   }
}
```

2. **main()**'de soldaki ağacın **root**u ile çağrıldığı varsayılan **traverse()** fonksiyonunun çıktısı nedir? **(25P)**

8 4 2 1 3 6 5 7 12 10 9 11 14 13 15

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

  DoublyNode* current = header;

  while (current->next != trailer)
  {
     if(newNode->score >= current->next->score)
        current = current->next;
     else
        break;
  }

  newNode->next       = current->next;
  newNode->prev       = current;
  ..............      = ..............;
  ..............      = ..............;
}

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);

}
```

3. **insertOrdered()** fonksiyonundaki **.....** satırları için aşağıda verilen kodlardan hangisi listeye hatalı ekleme yapar? **(25P)**

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

(A)
```
newNode->prev->next       = newNode;
newNode->next->prev       = newNode;
```

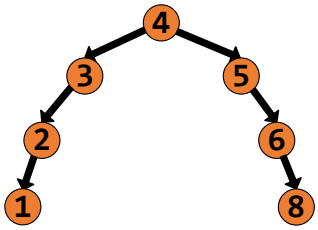
(B)
```
newNode->next->prev       = newNode;
newNode->prev->next       = newNode;
```

(C)
```
current->next             = newNode;
current->next->next->prev = newNode;
```

(D)
```
current->next             = newNode;
newNode->next->prev       = newNode;
```

(E)
```
current->next             = newNode;
current->next->prev       = newNode;
```

Zig	(X:Sol)
Zig-Zig	(X:Sağ, P:Sağ)
Zig-Zag	(X:Sağ, P:Sol)
Zig	(X:Sol)
Zig-Zag	(X:Sağ, P:Sol)
Zig-Zig	(X:Sol, P:Sol)
Zig-Zig	(X:Sağ, P:Sağ)
Zig-Zag	(X:Sağ, P:Sol)



4. Yukarıdaki işlemlerle oluşturulan Splay Ağacına verilerin hangi sırada eklendiğini bulunuz. **(25P)**

→ | 6 | 1 | 8 | 3 | 2 | 5 | 4 |