



CEVAPLAR

```
void print(DoublyNode* node, bool first)
{
    if (first)
        cout << node->elem << endl;
    if (node->next != trailer)
        print(node->next, false);
    else
        cout << node->elem << endl;
}

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, true);
}
```

1. Yukarıdaki programın çıktısı nedir? (25P)

Jack
Mike

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

    stl_stack.push(v);

    while (!stl_stack.empty())
    {
        Node* current = stl_stack.top();

        if ((current->right == NULL)
            && (current->left == NULL))
            cout << current->elt << " ";

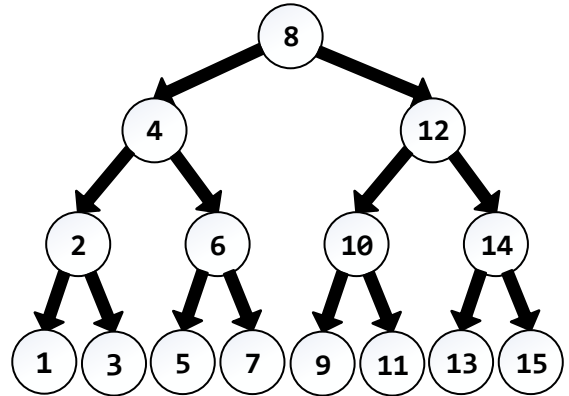
        stl_stack.pop();

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

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

2. main()'de aşağıdaki ağacın rootu ile çağrıldığı varsayılan traverse() fonksiyonunun çıktısı nedir? (25P)

1 3 5 7 9 11 13 15



```

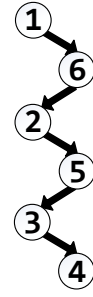
void insertOrdered(DoublyNode* newNode,
                  DoublyNode* current)
{
    if(..... && .....)

        insertOrdered(newNode, current->next);
    else
    {
        newNode->next = current->next;
        newNode->prev = current;
        current->next->prev = newNode;
        current->next = newNode;
    }
}

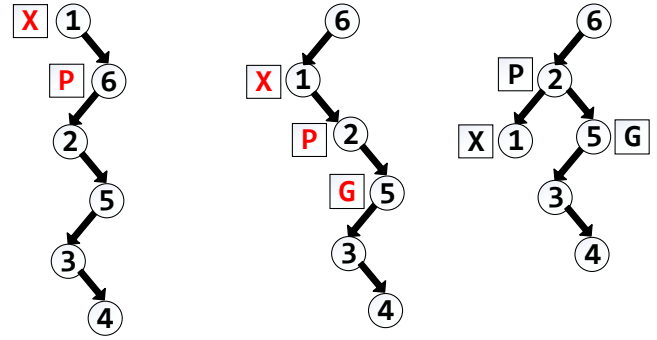
int main()
{
    DoublyLinkedList list; DoublyNode* newNode;
    newNode = new DoublyNode;
    newNode->elem = "Paul"; newNode->score = 720;
    list.insertOrdered(newNode, list.header);
    newNode = new DoublyNode;
    newNode->elem = "Rose"; newNode->score = 590;
    list.insertOrdered(newNode, list.header);
    newNode = new DoublyNode;
    newNode->elem = "Anna"; newNode->score = 660;
    list.insertOrdered(newNode, list.header);
    newNode = new DoublyNode;
    newNode->elem = "Mike"; newNode->score = 1105;
    list.insertOrdered(newNode, list.header);
}

```

Zig
Zig-Zig
Zig-Zig
Zig
Zig-Zig
Zig-Zig
Zig



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



3. insertOrdered() fonksiyonunu tamamlayınız. (25P)
Not → Trailer'ın score değerini 0 varsayınız.
Yanlış cevaptan 5P kılacaktır.

- (A) if ((newNode->score >= current->score) && (current != trailer))
- (B) if ((newNode->score >= current->next->score) && (current != trailer))
- (C) if ((newNode->score >= current->score) && (current->next != trailer))
- (D) if ((newNode->score >= current->next->score) && (current->next != trailer))

