Elif SAKAL – 200403623 Veri Yapıları ve Algoritmalar Ödevi

```
char dizi[boyut];
char stack[boyut];
int top = -1;
char eleman;
```

Burada öncelikle iki dizi tanımlıyorum ve boyutunu public olarak tanımlıyorum. Sonrasında stack'te yerimizi veren top=-1 tanımladım. Sonra ise stack'e eklenen parantezleri tutan veri değişkenini tanımladım. ----------
---

-

```
void push(char veri){
    if(top == boyut-1){
        puts("\n Üzgünüm stack doldu");
    }else{
        top++;
        stack[top] = veri;
    }
}
```

Burada stack içine veri ekleyen push fonksiyonunu yazdım. Öncelikle stack'in içinin top = boyut-1 olması durumunda stack'in full dolu olduğunu gösterdi. Aksi durumda ise stack'in içine veriyi ekleyip top değerini bir arttırdı. ----------
---

```
char peek(){
    return stack[top];
}
```

Burada stack'in top değerini döndüren fonksiyonu yazarız. Çünkü son girilen ve eşleşemeyen(varsa) parantezi bize göstermeli. ----------
---

```
char pop(){
    data = stack[top];
    top--;
}
```

Burada stack içinden eleman silen pop fonksiyonunu yazdım. Pop fonksiyonu stack'in içinden en üstteki elemanı siliyor ve top değerini bir azaltıyor.----------
---

```
bool bosmu(){
    if(top ==-1 ){
        return true;
    }else{
        return false;
    }
}
```

Burada stack'in içinin boş olup olmadığını kontrol eden bir fonksiyon yazdım. Bu fonksiyon ayrıca underflow olayını da engellemek için yazdım. Eğer

stack top değeri -1 ise true döner, eğer değilse false döner yani boş değildir.----------
---

```
void yazdir(){
    int j;
    for(j = 0; j<=top; j++){
        printf("\nStack : %c", stack[j]);
    }
}
```

Burada for döngüsü ile stack içindekilerin yazdırılmasını sağladım.----------
---

```
    int i;
    while(dizi[i] != '\0'){
        char d = dizi[i];
        printf("\nveri; %c", d);
        char peekd = peek();
```

- Burada while döngüsü ile önce dizinin içinin boş olmamasını kontrol ediyorum. Eğer boş değilse veriyi ve top değerinin indexini veriyor ve altta gelen if ler ile push pop yapıyorum.----------
---

```
        if(d== '(' || d== '{' || d== '['){
            push(d);
            printf("\npush edilen: %c", d);
        }
        else if(d== ')' || d== '}' || d== ']'){
            if(bosmu() == true){
                puts("\nstack bostur..");
                return false;
            }

            else if(d == ')' && peekd== '('){
                pop();

                printf("\npop edilen: %c", '(');
            }
            else if(d == '}' && peekd== '{'){
                pop();

                    printf("\npop edilen: %c", '{');
            }
            else if(d == ']' && peekd== '['){
                pop();

                    printf("\npop edilen: %c", '[');
            }else{
                return false;
            }
        }
        i+=1;
    }
    return bosmu();
```

Burada öncelikle açık parantez şeklinde veri girilyorsa bunları push fonksiyonu ile stack'e ekliyorum. Sonra eğer kapanan parantez ise yeni bir if ile boşmu fonksiyonu ile boş olup olamamasını kontrol ediyorum. Boşsa false değerini dönüyor.

Boş değilse else if'e gelince '()' şeklindeki parantezleri birbiri ile eşleştiriyor ve stack'in içinden pop edip çıkartıyor. Sonra bu işlemin aynısını '{}' şeklindeki parantez türü için yapıyor. En son ise '[]' şeklindeki parantez türü için yine aynı işlemleri yapıyor. Bu if ve else if lerin her birini yaptıktan sonra pop edilen parantezi bize gösteriyor. Bu şekilde her parantezin eşini bularak stack'ten tek tek çıkartıyor. En son ise her seferinde while döngüsünü 1 tane artırıyor tüm elemanlar için döngüyü devam ettirtiyor. Ve sonda boş olup olmadığını boşmu fonksiyonu ile döndürüyor.

```
int main(int argc, char *argv[]){

    puts("\nDegerleri giriniz");
    fgets(dizi, sizeof(dizi), stdin);
    yazdir();

if(dogruMu()== true){
    puts("\n Parantezler eşleşti");
}else{
    puts("\n Parantezler eslesmedi");
}
yazdir();
}
```

Burada int main fonksiyonunun içinde bizden alacağı parantezleri fgets ile dizinin içine alıyorum. Sonra yazdır fonksiyonu ile stack içindeki elemanları yazdırıyorum. Son olarak ise if ile doğrumu fonksiyonundaki parantez eşleştirmeli yapıp doğrumu fonksiyonu true ise yani tüm parantezler eşleşiyorsa eşleşti mesajı veriyorum. Eğer eşleşmediyse eşleşmedi mesajı verip stackte kalan elemanları yazdır fonksiyonu ile yazdırıyorum.

Bu şekilde bu kod ile stack'te parantez eşleştirme işlemlerini yapıyorum.