

## Aufgabe 10

Gegeben sei folgende Grammatik:

S: AB

A: aaA|B

B: b

Erstellen Sie einen Top-Down Parser nach der in der Vorlesung vorgestellten Methode des rekursiven Abstiegs in C++. Als Rahmen können Sie die Datei uebung9-rahmen.cpp benutzen. Sie enthält die benötigten Funktionen check(char c), scanner(), parser() und main(). Die explizit deklarierten Funktionen A(), B() müssen noch implementiert werden, sonst funktioniert der Parser nicht!

Im folgenden die Implementierungen für S() A() B() und C():

```
void S()
{
    printf("regel S->AB angewandt\n");
    A();B();
}

void A()
{
    if (token == 'a')
    {
        printf("regel A->aaA angewandt\n");
        check('a');check('a');
        A();
    }
    else
    {
        printf("regel A->B angewandt\n");
        B();
    };
}

void B()
{
    if (token == 'b')
    {
        printf("regel B->b angewandt\n");
        check('b');
    }
    else
    {
        printf("Syntaxfehler, 'b' erwartet\n");
    }
}
```

## Aufgabe 11

Gegeben sei folgende Grammatik:

S: ABC

A: aB|b

B: bb|C

C: c

Erstellen Sie einen Top-Down Parser nach der in der Vorlesung vorgestellten Methode des rekursiven Abstiegs in C++.

Im folgenden die Implementierungen für S() A() B() und C():

```
void S()
{
    printf("regel S->ABC angewandt\n");
    A();B();C();
}

void A()
{
    if (token == 'a')
    {
        printf("regel A->aB angewandt\n");
        check('a');
        B();
    }
    else
    {
        printf("regel A->b angewandt\n");
        check('b');
    }
};

void B()
{
    if (token == 'b')
    {
        printf("regel B->bb angewandt\n");
        check('b');check('b');
    }
    else
    {
        printf("regel B->C angewandt\n");
        C();
    }
}

void C()
{
    if (token == 'c')
    {
        printf("regel C->c angewandt\n");
        check('c');
    }
    else
    {
        printf("Syntaxfehler 'c' erwartet\n");
    }
}
```