Aufgabe 1:

Zeitaufwand: 30min

Einfach bruteforce für jeden char in s1 alle chars in s2 durchgehen und jedesmal bei dem man einen größeren gemeinsamen substring gefunden hat den speichern, bis man durch ist.

Laufzeitkomplexität:

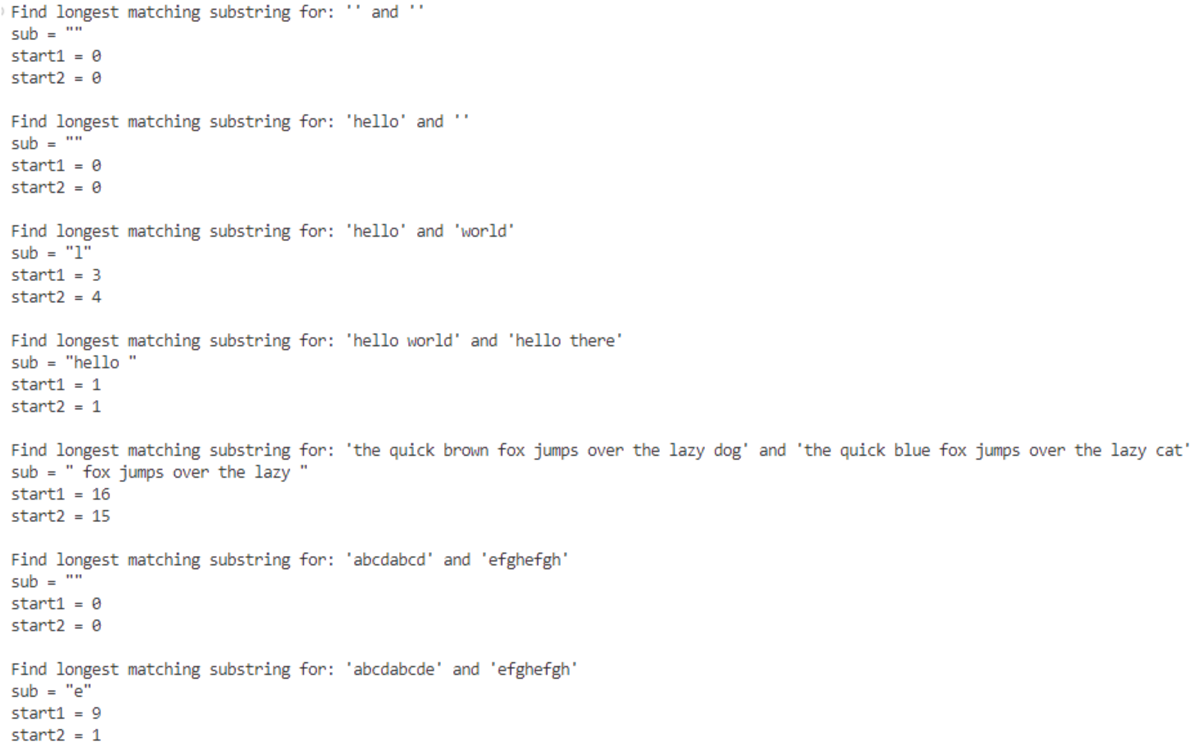
> worst case:
> O(m*n)
> m = length(s1)
> n = length(s2)

Code:

```
program FindLongestMatchingSubstring;

procedure FindLongestMatch(s1, s2: STRING; var sub: STRING; var start1,
start2: INTEGER);
var
  i, j, s1Len, s2Len, len, maxlen: INTEGER;
begin
  // init values
  s1Len := LENGTH(s1);
  s2Len := LENGTH(s2);
  maxlen := 0;
  sub := '';
  start1 := 0;
  start2 := 0;

  for i := 1 to s1Len do
    for j := 1 to s2Len do
      if s1[i] = s2[j] then
      begin
        len := 1;
        // find matching substring
        while (i + len - 1 <= s1Len) and (j + len - 1 <= s2Len) and (s1[i +
len - 1] = s2[j + len - 1]) do
          INC(len);
        // if found matching substring is longer than current longest found
matching substring save it
        if (len - 1) > maxlen then
        begin
          maxlen := len - 1;
          sub := COPY(s1, i, maxlen);
          start1 := i;
          start2 := j;
        end;
      end;
end;
```

```pascal
procedure TestFindLongestMatch(s1, s2: STRING);
var
  sub: string;
  start1, start2: integer;
begin
  WriteLn('Find longest matching substring for: ''', s1, ''' and ''', s2,
'''');
  FindLongestMatch(s1, s2, sub, start1, start2);
  WriteLn('sub = "', sub, '"');
  WriteLn('start1 = ', start1);
  WriteLn('start2 = ', start2);
  WriteLn;
end;

begin
  // Test Case 1: Both strings are empty
  TestFindLongestMatch('', '');

  // Test Case 2: One string is empty
  TestFindLongestMatch('hello', '');

  // Test Case 3: No common substring
  TestFindLongestMatch('hello', 'world');

  // Test Case 4: Common substring is at the beginning
  TestFindLongestMatch('hello world', 'hello there');

  // Test Case 5: Common substring is in the middle
  TestFindLongestMatch('the quick brown fox jumps over the lazy dog', 'the
quick blue fox jumps over the lazy cat');

  // Test Case 6: Multiple common substrings of the same length
  TestFindLongestMatch('abcdabcd', 'efghefgh');

  // Test Case 7: Multiple common substrings of different lengths
  TestFindLongestMatch('abcdabcde', 'efghefgh');
end.
```

Ergebnis Tests:



```
Find longest matching substring for: '' and ''
sub = ""
start1 = 0
start2 = 0

Find longest matching substring for: 'hello' and ''
sub = ""
start1 = 0
start2 = 0

Find longest matching substring for: 'hello' and 'world'
sub = "l"
start1 = 3
start2 = 4

Find longest matching substring for: 'hello world' and 'hello there'
sub = "hello "
start1 = 1
start2 = 1

Find longest matching substring for: 'the quick brown fox jumps over the lazy dog' and 'the quick blue fox jumps over the lazy cat'
sub = " fox jumps over the lazy "
start1 = 16
start2 = 15

Find longest matching substring for: 'abcdabcd' and 'efghefgh'
sub = ""
start1 = 0
start2 = 0

Find longest matching substring for: 'abcdabcde' and 'efghefgh'
sub = "e"
start1 = 9
start2 = 1
```