Aluno: Lenington do C. Rios, 13211187

1 StringMatching

1.1 Força bruta

Algorithm 1: naive(R,S)

input : A String R and a character sequence S
output: *true* if S is contained in R, or *false* otherwise

```
1  for i ← 0 to N − |S| + 1 do
2  |  j ← 0;
3  |  while j < |S| and R[i + j] = S[j] do
4  |  |  j ← j + 1;
5  |  |  if j = |S| then
6  |  |  |  return true;
7  |  |  end
8  |  end
9  end
10 return false;
```

1.2 Knuth-Morris-Pratt (KMP)

Algorithm 2: kmp(R,S)

input : A String R and a character sequence S
output: *true* if S is contained in R, or *false* otherwise

```
1  table ← createTable(S);
2  j ← 0;
3  for i ← 0 to |R| do
4  |  while R[i] = S[j] do
5  |  |  if j = |S| then
6  |  |  |  return true;
7  |  |  end
8  |  end
9  |  while R[i] ≠ S[j] do
10 |  |  if j > 0 then
11 |  |  |  j ← table[j − 1];
12 |  |  end
13 |  end
14 |  j ← j + 1;
15 end
16 return false;
```

1.3 Boyer-Moore-Horspool (BMH)

Algorithm 3: bmh(R,S)

input : A String R and a character sequence S
output: *true* if S is contained in R, or *false* otherwise

```
1  HashTable H ← createHashTable(S);
2  j ← 0;
3  t ← |S| − 1;
4  i ← t;
5  while i < |R| do
6  |   j ← |S| − 1;
7  |   while j ≥ 0 and R[i] = S[j] do
8  |   |   j ← j − 1;
9  |   end
10 |   if j = −1 then
11 |   |   return true;
12 |   end
13 |   if R[t] in H then
14 |   |   t ← t + H[R[t]];
15 |   else
16 |   |   t ← t + H["*"];
17 end
18 return false;
```