Task:

Programming Task

Implement a program to evaluate if a string matches a pattern. The program shall be executable from the command line. The program takes two arguments, the string to match and the pattern. The program prints “match” in case the pattern matches the string and “no match” in case the pattern does not match.

The glob pattern syntax is:

- ? matches any one character
- + matches one or more characters
- * matches zero or more characters
- \ escapes

Rules:

- Use the programming language of your choice
- Do not use eval
- Do not convert to regular expression
- Do not call a system glob function.

Keep in mind when writing the code that it should be maintainable by multiple people in the team.

Share your code by e-mail with instructions on how to compile and run.

Solution:

Question: The real question of this task is that with the the string S and pattern P, give e,f – does the pattern P[0:f] accept the string S[0:e]? When f at the end of pattern **P** and e at the end of string **S,** that is the result.

(1):

e-th f-th

S: a P: a

If these two patterns [ ] and [ ] match, P accepts S since S[e] == P[f].
So, the logic is: **(s[e] p[f]) and (s[e-1] p[f-1]) (1)**

Example: str = 'a l b e d c' and pat= 'a ? b * c' and the table M(i is y-axis, j is x-axis)

	" "	a	?	b	*	c
" "	T	F	F	F	F	F
a	F	T	F	F	F	F
l	F	F	T	F	F	F
b	F	F	F	T	T	F
m	F	F	F	F	T	F
n	F	F	F	F	T	F
c	F	F	F	F	T	T

- Case 1: when pat = " " and str = " ", M[0][0] = True
- Case 2: when pat = " " and str != " ", M[i][0] = False
- Case 3: when pat != " " and str = " ", M[0][j] = M[0][j-1] if p[j-1] == '*'

  Note: j-1 because we added an empty string column.

- Case 4: when p[j-1] == '?' or p[j-1] == s[i-1], M[i][j] = M [i-1][j-1] because of (1)
- Case 5: when p[j-1] == '*', there is two case:
  a. '*' represents null character: M[i][j] = M[i][j-1] because we ignore the * and check whether or not the previous pattern accept the string
  b. '*' represents non-null character: M[i][j] = M[i-1][j] because '*' consumed the i-th character in str

  M[i][j] = M[i][j-1] or M[i-1][j]

- Case 6: when p[j-1] = '\', it is the same of case 5a: M[i][j] = M[i][j-1]
- Case 7: when p[j-1] = '+', it is the same of case 5b and case '?':

  M[i][j] = M[i-1][j] or M [i-1][j-1]