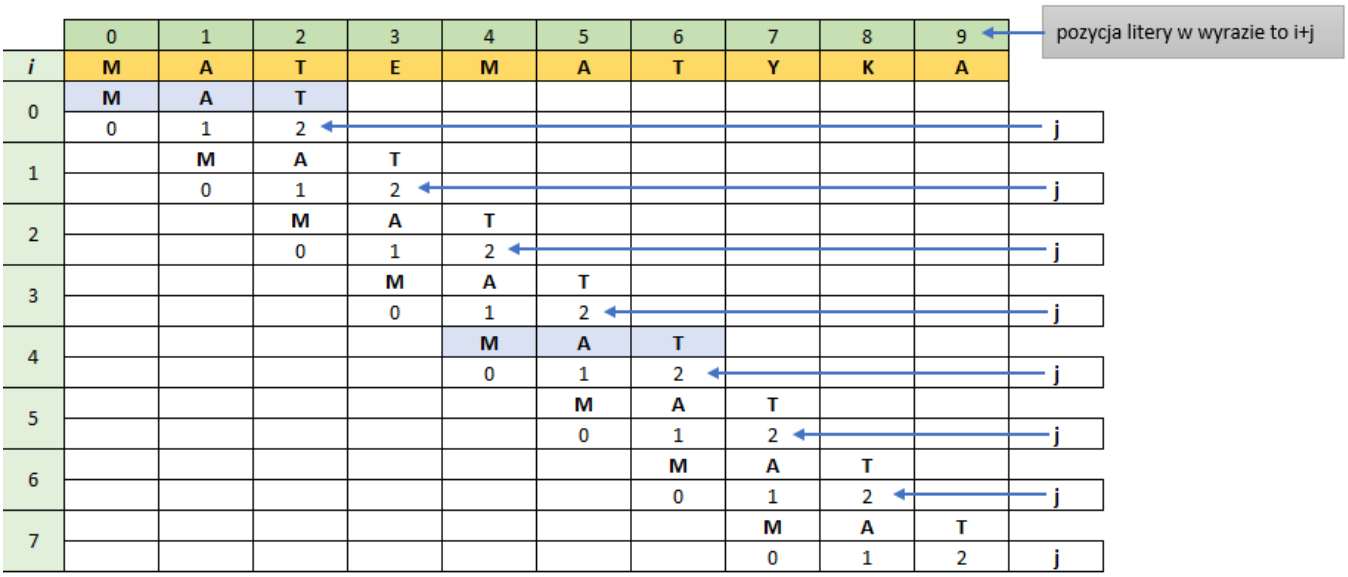
Podstawowy (naiwny) algorytm wyszukiwania wzorca w tekście.

1. Załóżmy, że mamy tekst: *AABBABABAABAABABAABA* i wzorzec: *ABAABA*.
2. **Należy odpowiedzieć na pytanie czy w podanym tekście znajdziemy wzorzec i jeśli TAK, to w których miejscach tego tekstu (od której pozycji znaku)?**
3. W powyższym przykładzie wzorzec występuje w tekście w 3 miejscach:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
A	A	B	B	A	B	***A***	B	A	***A***	B	A	A	B	***A***	B	A	A	B	A

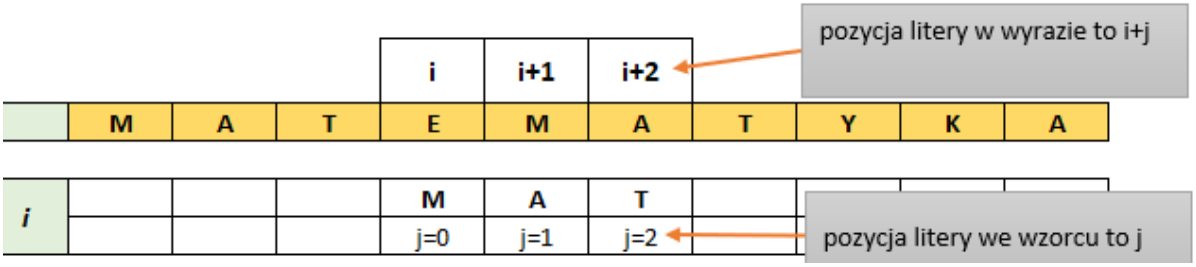
4. *Pierwsze rozwiązanie tego problemu, jakie przychodzi do głowy, to naiwny sposób sprawdzania dla każdej pozycji **i**, czy dany wzorzec występuje na tej pozycji.*
5. Dla tekstu: „*MATEMATYKA*” i wzorca: „*MAT*” algorytm ilustruje rysunek:



6. Na powyższym rysunku:
   a. i – pozycja w tekście od której zaczynamy sprawdzać występowanie wzorca
   b. j – pozycja znaku we wzorcu
7. Załóżmy, że d1 - długość tekstu, d2 - długość wzorca.
   Wtedy możemy stwierdzić, że:
   a. $i \in \{0, 1, \ \ldots, d1 - d2\}$
   b. $j \in \{0, 1, \ldots d2 - 1\}$
   c. Dla i, j jak wyżej sprawdzamy warunek: *tekst[i+j]==wzorzec[j] (lub tekst[i+j]!=wzorzec[j])*
8. W przykładzie z punktu 5:
   a. d1=10, d2=3, $i \in \{0, 1, 2, 3, 4, 5, 6, 7\}, j \in \{0, 1, 2\}$

Kodowanie.

1) Najpierw zajmiemy się sprawdzeniem czy wzorzec występuje na i-tej pozycji w tekście:



2) Utworzymy funkcję *czy_wzorzec()*, która:
   - zwraca liczbę -1 gdy od podanego *punktu_startu* w tekście wzorzec nie został znaleziony
   - funkcja zwraca liczbę *punkt_startu* jeśli od tej pozycji w tekście wzorzec został znaleziony
   - ma 3 argumenty:
     - tekst → string
     - wzorzec → string
     - punkt_startu → int

3) Przykładowy kod funkcji:

```
int czy_wzorzec(string tekst, string wzorzec, int punkt_startu){
    int j;
    for(j=0; j<wzorzec.size(); j++){
        if(tekst[punkt_startu+j]!=wzorzec[j]) return -1;
    }
    return punkt_startu;
}
```

4) Działanie funkcji:
   - Polecenie:

     ```
     cout << czy_wzorzec("matematyka", "mat", 1);
     ```

     daje w wyniku: `-1` (od znaku na pozycji 1 (drugi znak tekstu - a) nie znaleziono wzorca)
   - Polecenie:

     ```
     cout << czy_wzorzec("matematyka", "mat", 4);
     ```

     daje w wyniku: `4` (od znaku na pozycji 4 (piąty znak tekstu - m) jest wzorzec)

5) Teraz wystarczy wywołać funkcję *czy_wzorzec()* dla wszystkich możliwych punktów startu w tekście.

6) Przykładowy kod i działanie programu:

```cpp
//naiwne wyszukiwanie wzorca w tekście
#include <iostream>
using namespace std;

int czy_wzorzec(string tekst, string wzorzec, int punkt_startu){
    int j;
    for(j=0; j<wzorzec.size(); j++){
        if(tekst[punkt_startu+j]!=wzorzec[j]) return -1;
    }
    return punkt_startu;
}

int main() {
    string tekst;
    string wzorzec;
    int i, d1, d2, wynik;
    cout << "podaj tekst: ";
    getline(cin, tekst);
    cout << "podaj wzorzec: ";
    getline(cin, wzorzec);
    cout << "w wyrazeniu: " << tekst << endl;
    cout << "wzorzec: " << wzorzec << endl;
    cout << "zaczyna sie od pozycji nr: " << endl;
    d1=tekst.size();
    d2=wzorzec.size();
    for(i=0; i<=d1-d2; i++){
        wynik=czy_wzorzec(tekst, wzorzec, i);
        if(wynik>=0) cout << wynik << " ";
    }
    return 0;
}
```

```
w wyrazeniu: matematyka i informatyka to magia
wzorzec: ma
zaczyna sie od pozycji nr:
0 4 18 28
```

UWAGA:

W informatyce istnieją lepsze algorytmy szukania wzorca w tekście … 😊