ALGORYTMY I STRUKTURY DANYCH – BINARNE DRZEWO WYSZUKIWAŃ (BST)

- **BST –** drzewo binarne, w którym:
  - dla każdego węzła y w lewym poddrzewie pewnego węzła x spełnione jest ***key(y) < key(x)***, oraz
  - dla każdego węzła z w prawym poddrzewie pewnego węzła x spełnione jest ***key(z) > key(x)***

Operacje na BST

- reprezentacja dowiązaniowa

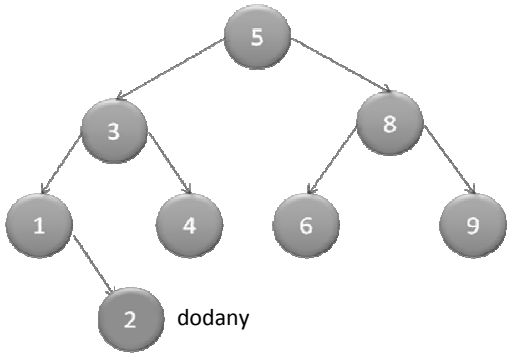
- wyszukiwanie węzła
  - **dane**: drzewo BST i wartość s
  - **wynik**: węzeł x taki, że key(x) = s lub NILL

```
x←root
bstSearch(x, s)
if(x = NILL or key[x] = s)
  ret(x)
if(s < key[x])
    ret(x←bstSearch(left[x],s)
else
   ret(x←bstSearch(right[x],s)
```

- **złożoność**: O(log(n)) (dla drzewa wyważonego)

- wstawianie węzła (węzeł jest zawsze dodawany jako liść)
  - **dane**: drzewo BST i węzeł y z wartością klucza s
  - **wynik**: drzewo BST z węzłem y

```
if(root = NILL)
  root←y
else
  x←root, p←x
  while(x ≠ NILL)
      p ← x
     if(key[y] < key[x])    x←left[x]
     else    x←right[x]
  if(key[y] < key[p])    left[p]←y
 else     right[p] ←y
```



- **złożoność**: O(log(n)) (dla drzewa wyważonego)

- usuwanie węzła
  - **dane**: drzewo BST i węzeł y z wartością klucza s (do usunięcia)
  - **wynik**: drzewo BST bez węzła y i węzeł y

  - przypadek 1 – węzeł nie posiada synów (proste usunięcie)
    - przypadek 2 – węzeł posada jednego syna – istniejący syn y zastępuje y
  - przypadek 3 – węzeł posiada obydwu synów
    - *wyznacz z, taki że:*
      - → *z jest następnikiem y, lub*
      - → *z jest poprzednikiem y*
    - *zastąp y przez z*
    - *usuń z (według przypadku 1 lub 2)*