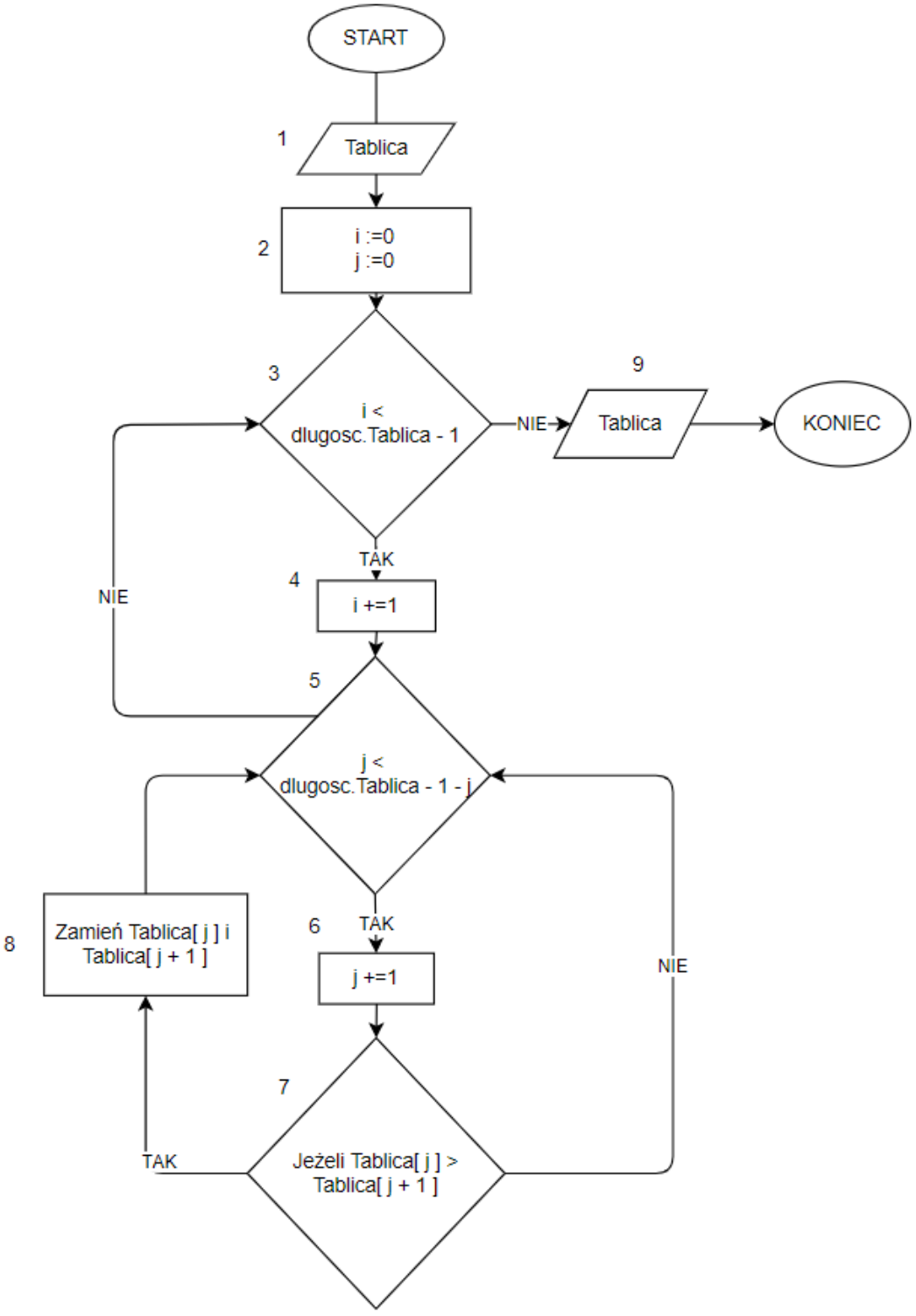
Lista 3

Sebastian Bednarski, 261662
24 kwietnia 2021 r.

Zad. 1

Projekt algorytmu- schemat blokowy sortowania bąbelkowego.



Analiza stanu dla tablicy [2, 1]

krok	tablica	i	j	i < dlugosc.Tablica - 1	j < dlugosc.Tablica - 1 - j	Tablica[j]	Tablica[j+1]	Tablica[j]>Tablica[j+1]	komentarz
1	[2,1]	-	-	-	-	-	-	-	wprowadzenie Tablicy
2	[2,1]	0	0	-	-	-	-	-	i := 0, j:=0
3	[2,1]	0	0	TAK	-	-	-	-	sprawdzenie warunku 3
4	[2,1]	1	0	TAK	-	-	-	-	zwiększenie i o 1
5	[2,1]	1	0	TAK	TAK	-	-	-	sprawdzenie warunku 5
6	[2,1]	1	1	TAK	TAK	2	1	-	zwiększenie j o 1
7	[2,1]	1	1	TAK	TAK	2	1	TAK	sprawdzenie warunku 7
8	[1,2]	1	1	TAK	TAK	1	2	TAK	zamiana Tablia[j] z Tablica[j+1]
5	[1,2]	1	1	TAK	NIE	1	2	TAK	sprawdzenie warunku 5
3	[1,2]	1	1	NIE	NIE	1	2	TAK	sprawdzenie warunku 3
9	[1,2]	1	1	NIE	NIE	1	2	TAK	zwrot Tablicy, Koniec algorytmu

Sortowanie bąbelkowe jest złożone z dwóch pętli(pętla po pętli), złożoność obliczeniowa w gorszych wypadkach jest zatem kwadratowa i oznaczamy ją $O(n^2)$. Przypadki znajdują się w kodzie dla tablic o wymiarach >1 (z wyjątkiem ostatnich 2 przypadków w kodzie). To właśnie pusta tablica lub jednowymiarowa jest najlepszym przypadkiem kodu i ma złożoność liniową, określana jest przez sprawdzenie jednej pętli i oznaczamy ją $O(n)$. Z wykorzystaniem Debuggera w Visualu można zbadać podane przypadki, ile razy zostaną wykonane pętle i jaką mają złożoność poszczególne tablice. Algorytm w najgorszym przypadku, czyli wszystkie elementy do zamiany, wykona dlugosc.Tablica – 1 obrotów pętli. Również można do kodu dodać flagi, aby to zobaczyć.