3 Wyszukiwanie największej z trzech liczb

W tej części zajmiemy się wyszukiwaniem największej z trzech liczb. Algorytm prosty, choć jego optymalne rozwiązanie okazać się może nieintuicyjne. Zadanie nadaje się świetnie do prowadzenia lekcji w cyklu Kolba.

Wszystkie ilustracje i grafiki zawarte w dokumencie są opracowaniami autorskimi.

3.1 Opis algorytmu

Zadanie polega na tym, że użytkownik podaje trzy różne liczby (mogą być rzeczywiste). Zadaniem programu jest wskazanie największej z nich.

3.2 Analiza

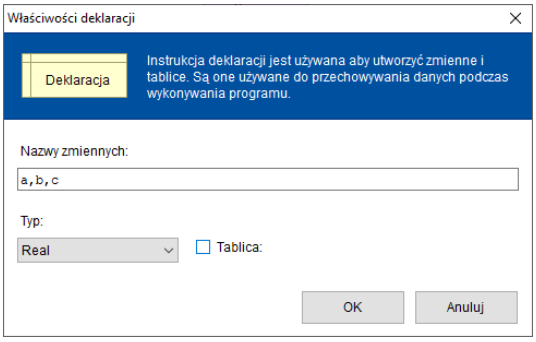
Zauważmy, że liczba jest największa spośród trzech wtedy gdy jest większa od dwóch pozostałych. Nazwijmy nasze liczby „a”, „b” oraz „c”. Doprowadza nas to do stworzenia trzech podwójnych warunków, które sprawdzimy kaskadowo:

1. $a \geq b$ oraz $a \geq c$,
2. $b \geq a$ oraz $b \geq c$,
3. $c \geq a$ oraz $c \geq b$ → jeśli dwa poprzednie nie są prawdziwe, ostatni warunek nie musi być sprawdzany.

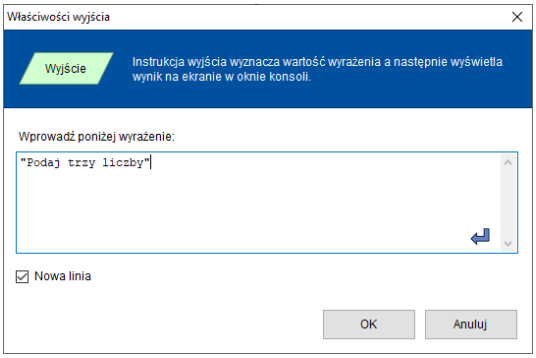
3.3 Budowa programu

Konstrukcja algorytmu jest właściwie natychmiastowa i większość uczniów potrafi podać poprawny opis po chwili zastanowienia.

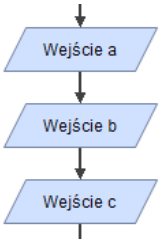
Deklarujemy niezbędne zmienne



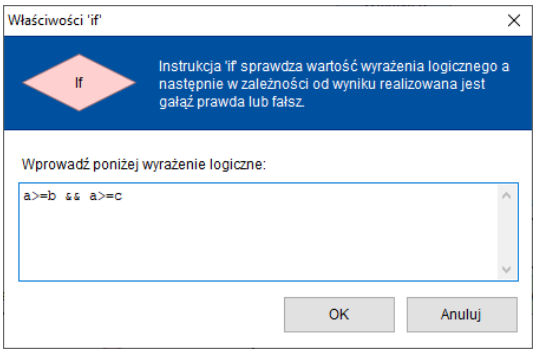
Dodajemy monit o podanie trzech liczb



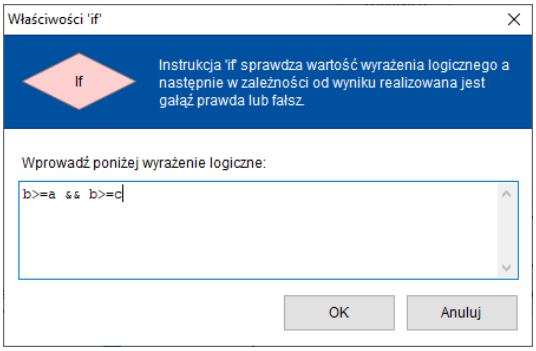
Pobieramy wartości



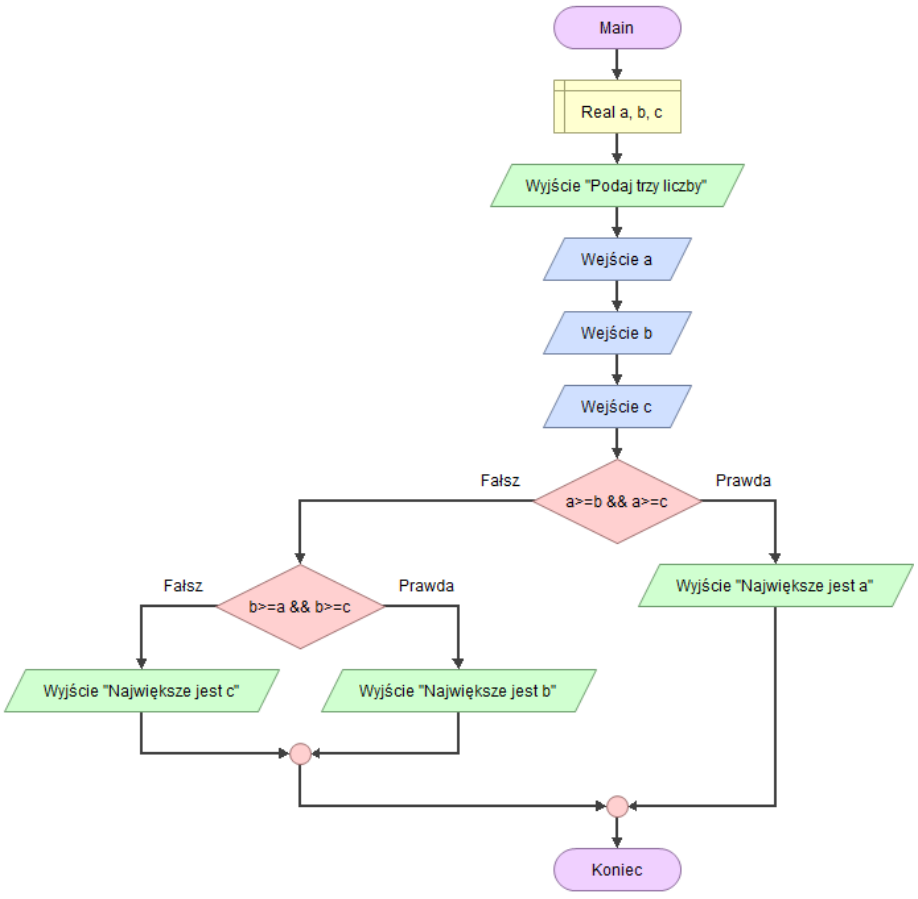
Warunki rozwiązujemy etapami rozpoczynając np. od zmiennej „a”. Jeżeli warunek jest prawdziwy, oznacza to, że zmienna „a” przechowuje największą wartość*.



Jeśli powyższy warunek jest nieprawdziwy sprawdzamy, czy największą liczbą jest może „b”.
Jeśli warunek jest prawdziwy, to faktycznie „b” przechowuje największą wartość



W przypadku, gdy warunek zwrócił „fałsz” oznacza to, że największa wartość jest przechowywana w ostatniej zmiennej → „c”. Kompletny algorytm wygląda następująco:



* W przypadku równych liczb, nierówność nieostra >= powoduje, że wybraną będzie ta wartość, którą sprawdzana później.

3.4 Optymalizacja

Powyższy algorytm jest merytorycznie poprawny jednak gdybyśmy spróbowali zrobić eksperyment myślowy, w którym chcemy odnaleźć największą liczbę spośród czterech, może pięciu…?

Szybko dojdziemy do wniosku, że liczba warunków i porównań w nich zawartych bardzo szybko rośnie, a samo „drzewko" zaczyna się mocno komplikować.

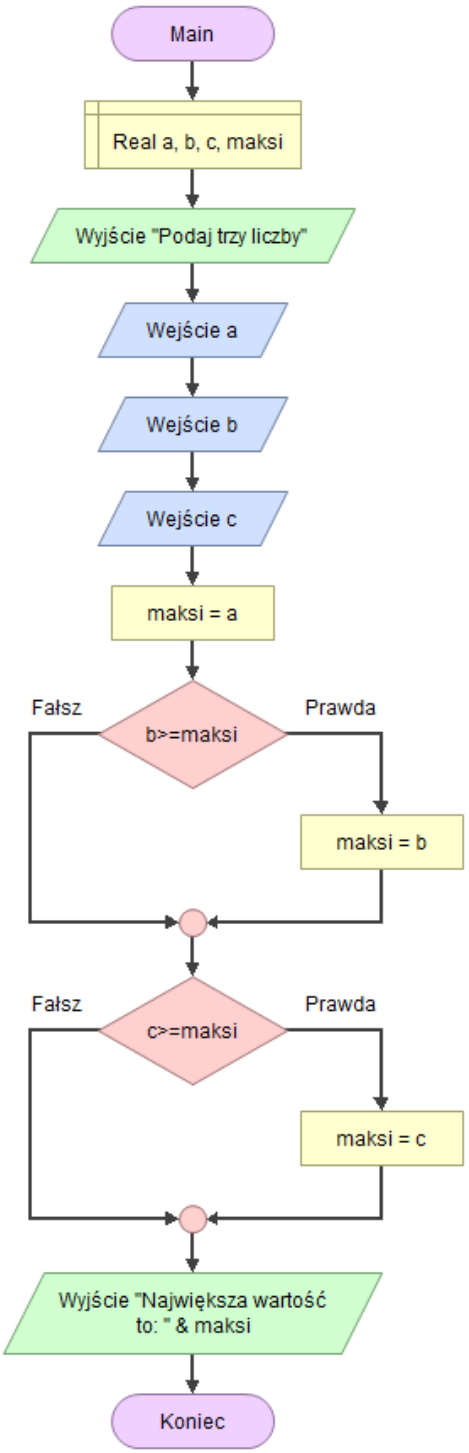
Rozwiązaniem, optymalizacją a zarazem krokiem w stronę wyszukiwania największej (lub najmniejszej) wartości w dowolnie dużym zbiorze danych jest inne podejście.

1. Zakładamy, że największą liczbą jest „a" (przypisujemy jej wartość do zmiennej pomocniczej „maksi"[*])
2. Sprawdzamy, czy „b" jest większe od zapamiętanej największej liczby. Jeśli jest to prawdą, to teraz „b" staje się naszą wartością największą.
3. Sprawdzamy, czy „c" jest większe od zapamiętanej największej liczby. Jeśli tak, to teraz „c" staje się wartością największą.

Przedstawiony algorytm pozwala zredukować liczbę porównań do dwóch oraz sprawia, że program staje się skalowalny (można go rozbudować do badania większej liczby zmiennych «lub całych zbiorów liczb»).

Obydwa programy można pobrać z mojej strony:

1. Wersja (intuicyjna):
   http://maczek.edu.pl/Tomasz_Bilinski/algorytmy/02_max_z_trzech_v-a.fprg
2. Wersja zoptymalizowana:
   http://maczek.edu.pl/Tomasz_Bilinski/algorytmy/02_max_z_trzech_v-b.fprg



[*] Unikamy słowa „max" ze względu na możliwe kolizje z nazwami funkcji w niektórych językach programowania