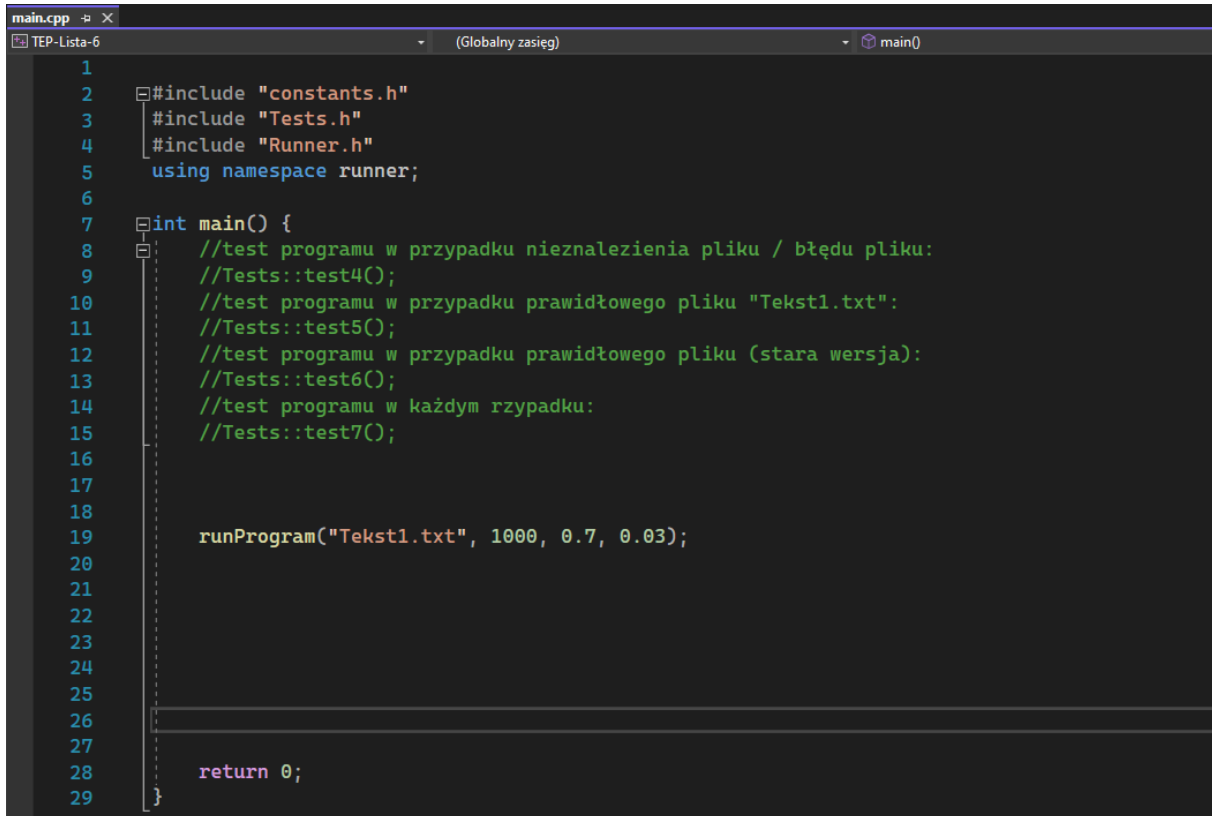


# Instrukcja obsługi algorytmu genetycznego

## Sposób uruchomienia:

W celu uruchomienia programu należy zmodyfikować wywołanie `runProgram()` w pliku `main.cpp` lub skorzystać z jednej z przygotowanych wcześniej procedur testowych.



```
1
2 #include "constants.h"
3 #include "Tests.h"
4 #include "Runner.h"
5 using namespace runner;
6
7 int main() {
8     //test programu w przypadku niezalezienia pliku / błędu pliku:
9     //Tests::test4();
10    //test programu w przypadku prawidłowego pliku "Tekst1.txt":
11    //Tests::test5();
12    //test programu w przypadku prawidłowego pliku (stara wersja):
13    //Tests::test6();
14    //test programu w każdym rzypadku:
15    //Tests::test7();
16
17
18
19    runProgram("Tekst1.txt", 1000, 0.7, 0.03);
20
21
22
23
24
25
26
27
28    return 0;
29 }
```

W celu uruchomienia jednej z procedur testowych (nr 4, 5, 6 lub 7) należy usunąć poprzedzające ją znaki komentarza ( // ). W celu normalnego uruchomienia programu z wykorzystaniem własnych ustawień należy natomiast zmodyfikować argumenty w wywołaniu funkcji `runProgram()`. Tymi argumentami są w kolejności: ścieżka do pliku tekstowego zawierającego dane problemu, zakładana wielkość populacji, prawdopodobieństwo/współczynnik krzyżowania osobników i prawdopodobieństwo/współczynnik mutowania osobników. Można również uruchomić program podając tylko część parametrów. Możliwe wywołania programu wyglądają następująco:

```
static void runProgram(string filePath, int pop, double cross, double mutate) { ... }
static void runProgram(string filePath) { ... }
static void runProgram(int pop, double cross, double mutate) { ... }
static void runProgram() { ... }
```

Przykłady:

```
runProgram("Tekst1.txt", 1000, 0.7, 0.03);
runProgram("Tekst.txt");
runProgram(1000, 0.7, 0.03);
runProgram();
```

## Sposób działania i wyświetlania wyników:

Po uruchomieniu programu wypisane zostaną szczegółowe dane programu. Na początek wypisana zostanie informacja, czy i który plik został załadowany, następnie wypisana jest wielkość plecaka (czyli tyle ile ciężaru można w nim unieść) oraz ilość przedmiotów możliwych do wybrania. Następnie w dwóch kolumnach wyświetlone zostają wszystkie przedmioty możliwe do spakowania w formacie: wartość przedmiotu po lewej stronie i waga przedmiotu po prawej stronie. Następnie program wyszukuje najlepsze rozwiązanie danego problemu i za każdym razem gdy znajdzie lepsze rozwiązanie, w sekcji „results:”, wypisuje nowe najlepsze znalezione rozwiązanie. Na koniec program wypisuje wartość najlepszego rozwiązania, oraz jego kod genetyczny (kod rozwiązania).

```
Konsola debugowania programu Microsoft Visual Studio
succeeded to load from file 'Tekst1.txt'!
problem details:

backpack size: 18
amount of items: 10
worth:         weight:
1             1
2             2
3             3
4             4
5             5
6             6
7             7
8             8
9             8
10            10

results:
found new best fitness: 0
found new best fitness: 8
found new best fitness: 12
found new best fitness: 15
found new best fitness: 16
found new best fitness: 19

best fitness achieved: 19
best genetic code achieved: 1010010010

C:\Users\Kuba\source\repos\TEP-Lista-6\x64\Debug\TEP-Lista-6.exe (proces 29060) zakończono z kodem 0.
Aby automatycznie zamknąć konsolę po zatrzymaniu debugowania, włącz opcję Narzędzia -> Opcje -> Debugowanie -> Automatycznie zamknij konsolę po zatrzymaniu debugowania.
Naciśnij dowolny klawisz, aby zamknąć to okno...
```