Zadanie 1 (WSI)

praca domowa 23.10.2020

Piotr Obst 304090 (grupa 103)

1. **Treść zadania**

   Temat 1.1
   Zadany jest zbiór N przedmiotów z odpowiednimi wagami (w kilogramach) i wartościami (w złotych). Dana jest maksymalna pojemność plecaka. Znaleźć podzbiór przedmiotów, które mieszczą się w plecaku i mają maksymalną wartość łączną. Zaimplementuj i zbadaj algorytm siłowy do szukania optymalnego zbioru oraz algorytm zachłanny. Przygotuj przykładowy zbiór testowy oraz porównaj liczbę iteracji oraz wyniki obydwu algorytmów dla tego zbioru.

2. **Sposób uruchomienia i ogólny opis programu**
   - Należy mieć zainstalowanego Pythona 3,
   - Aby uruchomić program, należy wpisać w konsoli „python main.py”,
   - Program wykonuje algorytm siłowy oraz zachłanny dla wszystkich plików z rozszerzeniem „txt” znajdujących się w folderze „input_files”,
   - Następnie wyświetlane jest porównanie obu algorytmów dla danego pliku w formie tabelki (przedmioty wymienione w tabeli to te, które zostały spakowane, a nie wszystkie dostępne),
   - Dla większej ilości przedmiotów należy chwilę poczekać na zakończenie działania programu.
3. **Dodatkowe funkcje**
   - Oprócz wymaganej obsługi pojedynczych przedmiotów dodałem również możliwość dodawania N takich samych przedmiotów. W takim przypadku należy w pliku wejściowym podać dane w formacie „<waga> <cena> <ilość>” (ilość jest opcjonalna),
   - Ponadto, można dodać nieskończoną ilość przedmiotów – wtedy wpisać należy „-1” jako ilość.
4. **Podsumowanie i wnioski**
   - Zgodnie z ustaleniami, starałem się pisać kod samodokumentujący się, więc raczej nie ma potrzeby opisywania kodu w tym raporcie,
   - Kod pisałem w oparciu o standard Pep8 oraz starałem się, aby był jak najbardziej uniwersalny i stabilny,
   - Algorytm siłowy zawsze zwraca poprawne wyniki, jednak bardzo szybko rośnie liczba operacji w miarę zwiększania liczby dostępnych przedmiotów,
   - Algorytm zachłanny jest szybki, jednak nie zawsze daje najbardziej optymalne wyniki,
   - Na następnej stronie znajduje się przykładowe wyjście programu:

```
Please wait, it can take some time...
```

file	input_files/basic.txt	
max weight	7	
algorithm	bruteforce	greedy
weight	7	7 equal
value	14	14 equal
iterations	15	4 less
items	Item(weight: 1, value: 2) Item(weight: 3, value: 4) Item(weight: 2, value: 5) Item(weight: 1, value: 3)	Item(weight: 1, value: 3) Item(weight: 2, value: 5) Item(weight: 1, value: 2) Item(weight: 3, value: 4)

file	input_files/basic_multiple.txt	
max weight	17	
algorithm	bruteforce	greedy
weight	15	15 equal
value	33	33 equal
iterations	4095	12 less
items	Item(weight: 1, value: 2, amount: 6) Item(weight: 3, value: 4) Item(weight: 2, value: 5) Item(weight: 1, value: 3, amount: 4)	Item(weight: 1, value: 3, amount: 4) Item(weight: 2, value: 5) Item(weight: 1, value: 2, amount: 6) Item(weight: 3, value: 4)

file	input_files/big.txt	
max weight	70	
algorithm	bruteforce	greedy
weight	65	65 equal
value	504	504 equal
iterations	255	8 less
items	Item(weight: 5, value: 60) Item(weight: 10, value: 100) Item(weight: 15, value: 120) Item(weight: 9, value: 12) Item(weight: 16, value: 22) Item(weight: 1, value: 80) Item(weight: 9, value: 110)	Item(weight: 1, value: 80) Item(weight: 9, value: 110) Item(weight: 5, value: 60) Item(weight: 10, value: 100) Item(weight: 15, value: 120) Item(weight: 16, value: 22) Item(weight: 9, value: 12)

file	input_files/big_multiple.txt	
max weight	333	
algorithm	bruteforce	greedy
weight	333	290 less
value	2486	2181 less
iterations	524287	17 less
items	Item(weight: 10, value: 100) Item(weight: 15, value: 120) Item(weight: 1, value: 80, amount: 7) Item(weight: 9, value: 110) Item(weight: 73, value: 399, amount: 4)	Item(weight: 1, value: 80, amount: 7) Item(weight: 9, value: 110) Item(weight: 5, value: 60) Item(weight: 10, value: 100) Item(weight: 15, value: 120) Item(weight: 73, value: 399, amount: 3) Item(weight: 16, value: 22) Item(weight: 9, value: 12)

file	input_files/greedy_wrong.txt