Algorytmy i struktury danych, Teleinformatyka, I rok

Raport z laboratorium nr: 5

Imię i nazwisko studenta: Amadeusz Gunia

1. W pole poniżej wklej najważniejszy (według Ciebie) fragment kodu źródłowego z zajęć (maksymalnie 15 linii).

```
1. for i in range(len(I)):
2.     s=W[i]*H[i]
3.     id=I[i]
4.     val=V[i]
5.     rat=val/s
6.     R.append([rat, id])
7. R.sort()
8. R.reverse()
9.
10.
11.
12.
13.
14.
15.
```

Uzasadnij swój wybór.

Uważam, że ten fragment kodu jest najważniejszy, gdyż oblicza pole zajmowane przed dany przedmiot oraz stosunek wartości tego przedmiotu do jego pola. Następnie tworzona jest lista takich stosunków (dla każdego przedmiotu). Posortowana malejąco lista jest potrzebna do dalszej części algorytmu.

2. Podsumuj wyniki uzyskane podczas wykonywania ćwiczenia. Co ciekawego zauważyłeś? Czego się nauczyłeś? Jeśli instrukcja zawierała pytania, odpowiedz na nie. Do sprawozdania możesz dodać wykresy jeśli jest taka potrzeba.

W tym ćwiczeniu wykonałem dwa algorytmy, które rozwiązują problem plecakowy. W obu algorytmach mechanizm działa w następujący sposób: dla iterowanego przedmiotu sprawdzane jest czy jego szerokość i wysokość są mniejsze lub równe aktualnym wymiarom plecaka. Jeśli przedmiot spełnia ten warunek, jest dodawany do plecaka, a wymiary plecaka zostają pomniejszone o długość i szerokość przedmiotu. W pierwszym algorytmie do plecaka wkładane są przedmioty w kolejności według listy dołączonej do ćwiczenia. Wyniki otrzymanych wartości dla poszczególnych wielkości plecaka dołączam poniżej jako zrzuty ekranu. Widać na nich również, że czas wykonania tego algorytmu jest bardzo krótki (milisekundy).

W drugim algorytmie, wkładane przedmioty są wybierane na podstawie stosunku wartości danego przedmiotu do jego pola powierzchni. Jako pierwszy zostaje wybrany przedmiot z największym stosunkiem, następnie kolejne przedmioty - według tej samej zasady. Dzięki temu uzyskujemy o wiele lepszy wynik niż w pierwszym algorytmie. Im większa pojemność plecaka tym widać lepszy efekt tego algorytmu. Czas wykonania tego algorytmu jest kilkukrotnie większy niż czas poprzedniego – co wynika z utworzenia dodatkowej tablicy stosunków i posortowania jej – ale nadal jest to czas (w wersji plecaka 1000x1000) nieprzekraczający 0,14 sekundy.

Poniżej dołączam zrzuty ekranu bezpośrednio z Pythona.