Projektowanie i Eksploatacja Sieci Komputerowych
Krotki, listy i zaawansowane grafy losowe

1. Skopiuj kod z poprzednich zajęć – będzie stanowił bazę dla kolejnego laboratorium.
2. Zmodyfikuj program z laboratorium 3 tak, aby zamiast na literach operował na numerach wierzchołków
3. Dodatkowo zmodyfikuj funkcję __init__ tak, aby domyślnie konstruktor nie akceptował żadnych parametrów
4. Czy zmiana z punktu 2 wpłynie na funkcję addVertex i funkcję dodającą krawędzie, czy będzie należało je zmodyfikować?
5. Przetestuj działanie poleceń (może być w nowym pliku):
   import sys
   print(sys.argv)
6. Jak polecenia z punktu 5 zadziałają, jeśli w wywołaniu skryptu (python nazwa_skryptu) dodasz argumenty wiersza poleceń? (np. Python nazwa_skryptu arg1 arg2 arg3)
7. Zmodyfikuj program z laboratorium 3 tak, aby akceptował jako pierwszy argument wiersza poleceń ilość wierzchołków, które następnie skrypt doda losowo do grafu. Przetestuj działanie polecenia, deklarując klasę z pustym grafem i dodając do niego losowo 20 wierzchołków i krawędzi. Kod programu załącz do sprawozdania
8. Dodaj do klasy funkcję vertexStep, która dla podanego wierzchołka obliczy jego stopień
9. Dodaj do klasy funkcję printVertexSteps, która w pętli obliczy i wypisze stopnie wszystkich wierzchołków grafu do konsoli
10. Dodaj do klasy pole edgeCount oraz funkcję countEdges, która policzy ilość krawędzi w grafie (pamiętaj że jest nieskierowany – krawędź od a do b i od b do a to ta sama krawędź) i zapisze ją do zmiennej edgeCount
11. Zmodyfikuj funkcje dodające losowe wierzchołki tak, aby automatycznie zwiększała edgeCount.
12. Kod programu załącz do sprawozdania
13. Dodaj do klasy zmienną tablicową steps.
14. Korzystając z możliwości budowania tablic wielowymiarowych oraz funkcji vertexStep wypełnij zmienną steps dwuelementowymi tablicami postaci [nr_wierzcholka, stopien]. Skorzystaj z konstrukcji:
    self.steps.append([nr_wierzcholka, stopien])
15. Korzystając z konstrukcji:
    self.posortowana = sorted(self.steps,key=lambda x: x[1])
    posortuj tablicę steps
16. Zmodyfikuj program w taki sposób, żeby w każdym nowym grafie, zadeklarowanym jako obiekt klasy, konstruktor dodawał 30 wierzchołków losowo. Następne n wierzchołków (n podane przez użytkownika w wierszu poleceń) powinien dodawać również losowo, ale z większym prawdopodobieństwem do wierzchołków o większym stopniu.
    Podpowiedź: szukając wierzchołka, do którego dołączyć nowy, losuj liczbę od 0 do k (gdzie k wynosi łączny stopień wszystkich wierzchołków w grafie, czyli 2 razy ilość krawędzi). Następnie sprawdzaj w pętli, czy wylosowana liczba jest mniejsza od kolejnych elementów tablicy self.posortowana. Jeśli jest, nowy wierzchołek dodaj do aktualnie sprawdzanego wierzchołka z posortowanej tablicy. Jeśli nie, odejmuj od wylosowanej liczby stopień aktualnie sprawdzanego wierzchołka i procedurę powtarzaj, aż do znalezienia odpowiedniego elementu. Zwróć uwagę na sytuacje, w których k = łącznemu stopniowi. W tym przypadku dodawaj nowy wierzchołek do ostatniego w tablicy self.posortowana Pamiętaj, że po dodaniu nowego wierzchołka należy znów zbudować tablicę self.steps oraz self.posortowana.
17. Kod programu załącz do sprawozdania