# Опис алгоритмів

Перелік всіх основних змінних та їхнє призначення наведено в таблиці 3.1.

Таблиця 3.1 – Основні змінні та їхні призначення

|  |  |
| --- | --- |
| Змінна | Призначення |
| Vertexes | Множина вершин |
| Edges | Множина ребер |
| Action | Дія активної кнопки |
| Current | Збереження ідентифікатора об’єкта на полотні, найближчого до вказівника миші |
| Max\_v | Вершина з найбільшою кількістю ребер |
| Edge | Ребро |
| Res | Список результату роботи метода |

## Загальний алгоритм

1. ПОЧАТОК
2. ПОЧАТОК ЦИКЛУ
3. ПОКИ працює вікно програми:
   1. ЯКЩО в змінній Action значення «вершина» і користувач натискає на полотно, ТО створити вершину на полотні і додати її до Vertexes.
   2. ЯКЩО в змінній Action значення «ребро»:
      1. ЯКЩО користувач вибирає на полотні дві раніше не зв’язані ребрами вершини, ТО створити на полотні ребро між вибраними вершинами та додати його до Edges.
      2. ЯКЩО користувач створює ребро з початком і кінцем в одній вершині, АБО користувач створює ребро між вершиною і ребром, АБО користувач створює ребро між ребрами, АБО користувач створює ребро між вже зв’язаними вершинами, ТО вивести на екран повідомлення про помилку.
   3. ЯКЩО в змінній Action значення «переміщення» і на об’єкт полотна з можливістю переміщення натиснута ЛКМ, ТО в об’єкті Current змінити значення координат на координати вказівника миші.
   4. ЯКЩО в змінній Action значення «видалення»:
      1. ЯКЩО натиснутий об’єкт Current належить Vertexes, ТО видалити з Edges всі ребра пов’язані з об’єктом Current, та видалити об’єкт Current з Vertexes.
      2. ЯКЩО натиснутий об’єкт Current належить Edges, ТО видалити об’єкт Current з Edges.
   5. ЯКЩО в змінній Action значення «approx» і натиснута кнопка «вирішити», ТО вивести на екран рішення алгоритму «approx».
   6. ЯКЩО в змінній Action значення «greedy» і натиснута кнопка «вирішити», ТО вивести на екран рішення алгоритму «greedy».
4. КІНЕЦЬ ЦИКЛУ
5. КІНЕЦЬ

## Алгоритм Approx-Vertex-Cover методу

1. ПОЧАТОК
2. ЯКЩО множина Edges пуста, АБО в множині Vertexes є вершини з кількістю ребер 0, ТО вивести на екран повідомлення про відсутність вхідних даних. ІНАКШЕ:
3. ПОЧАТОК ЦИКЛУ
4. ПОКИ множина Edges не пуста:
   1. Вибираємо випадкове ребро Edge з множини Edges.
   2. Додаємо вершини ребра Edge до списку Res.
   3. Видаляємо з множини Edges всі ребра пов’язані з вершинами ребра Edge.
5. КІНЕЦЬ ЦИКЛУ
6. Виводимо на екран результати методу Res.
7. КІНЕЦЬ

## Алгоритм жадібного (greedy) методу

1. ПОЧАТОК
2. ЯКЩО множини Vertexes і Edges пусті, ТО вивести на екран повідомлення про відсутність вхідних даних. ІНАКШЕ:
3. ПОЧАТОК ЦИКЛУ
4. ПОКИ множини Vertexes і Edges не пусті:
   1. Знаходимо вершину з найбільшою кількістю ребер Max\_v.
   2. Додаємо вершину Max\_v до списку Res.
   3. Видаляємо Max\_v з множини Vertexes.
   4. Видаляємо з множини Edges всі ребра пов’язані з вершиною Max\_v.
   5. ЯКЩО в множині Vertexes є вершини з кількістю ребер 0, ТО видаляємо їх з Vertexes.
5. КІНЕЦЬ ЦИКЛУ
6. Виводимо на екран результати методу Res.
7. КІНЕЦЬ