Альтернативный экзамен

по ДМ на тему

"Нахождение компонент двусвязности графа"

Студента 1 курса ФКТИ, гр.6306

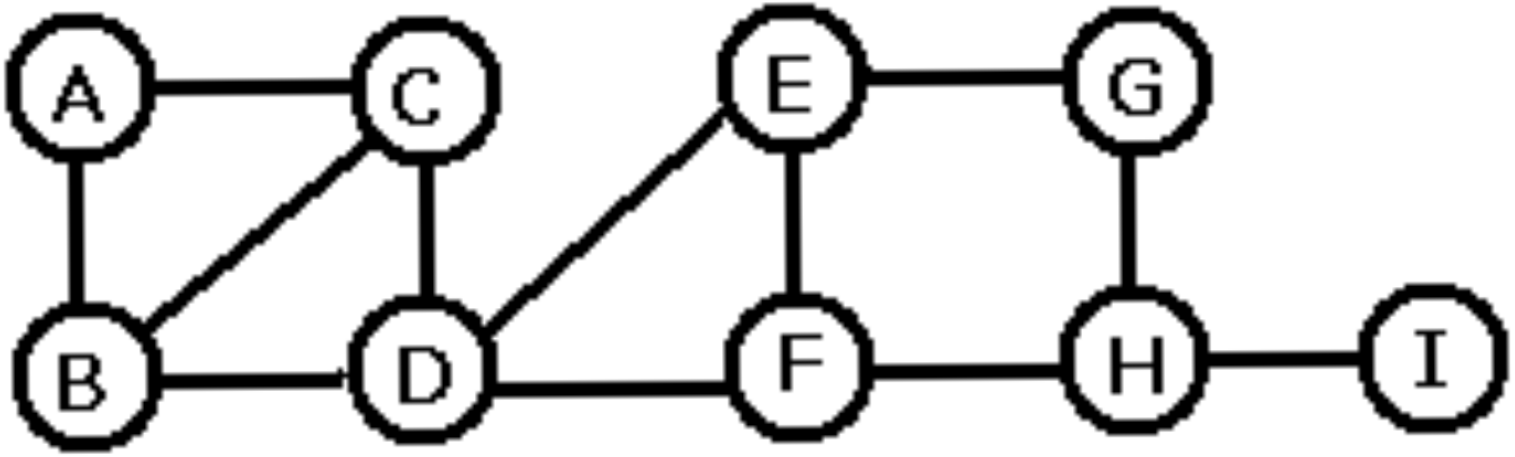
Солдатенкова Андрея

Задача:

- По книге Липского "Комбинаторика для программистов" изучить то,что связано с темой "Нахождение компонент двусвязности графа" (с.95-99). Разобраться с алгоритмом и алгоритмом, на котором он основан. Написать программу, находящую компоненты двусвязности и визуализировать её (представить исходный граф и его компоненты - для визуализации можно использовать библиотеки, и не писать программу самому)

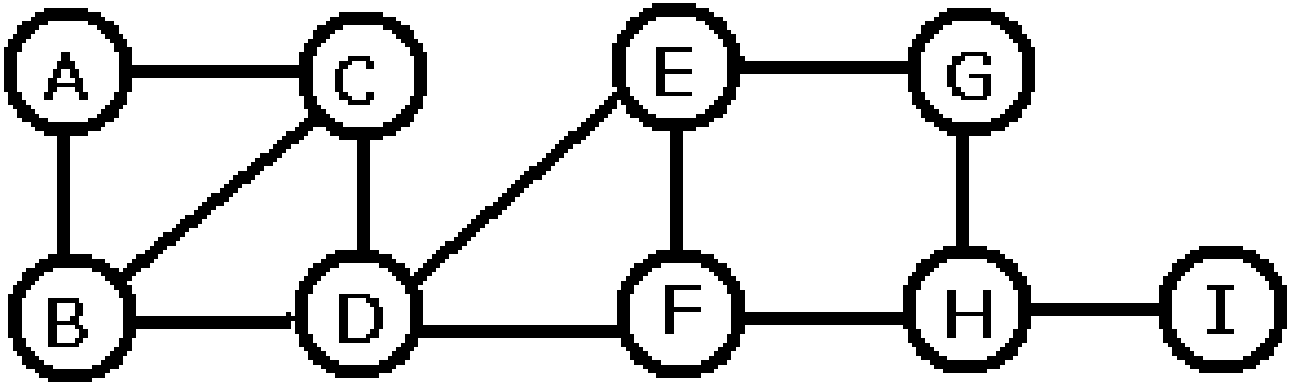
Компонентой двусвязности графа называется такое максимальное подмножество из трех или более его вершин, в котором любые две вершины соединены, по крайней мере, двумя путями, не имеющими общих ребер.

Компонента двусвязности - устойчивая часть графа: если в ней удалить вершину и все примыкающие к ней ребра, то любые две из оставшихся вершин по-прежнему оказываются соединенными между собой.



Пример графа с тремя компонентами двусвязности. Вершины первой компоненты имеют метки A, B, C, D; вершины второй - метки D, E, F, G; третья компонента содержит вершины H и I.

- Точками сочленения в связном графе служат такие вершины, удаление которых превращает граф в несвязный. Точки сочленения - это такие вершины, которые принадлежат сразу двум компонентам двусвязности.
- Ниже на примере это вершины D и H. Определение точек сочленения и компонент двусвязности тесно связаны между собой.



Алгоритм будет построен на алгоритме поиска в глубину

При обходе в глубину мы посещаем первый узел, а затем идем вдоль ребер графа, пока не упремся в тупик. Узел неориентированного графа является тупиком, если мы уже посетили все примыкающие к нему узлы. В ориентированном графе тупиком также оказывается узел, из которого нет выходящих ребер. После попадания в тупик мы возвращаемся назад вдоль пройденного пути пока не обнаружим вершину, у которой есть еще не посещенный сосед, а затем двигаемся в этом новом направлении. Процесс оказывается завершенным, когда мы вернулись в отправную точку, а все примыкающие к ней вершины уже оказались посещенными.

- **Нахождение компонент двусвязности.**

При реализации алгоритма будем подсчитывать число посещенных узлов графа. Каждому узлу сопоставим индекс - номер, указывающий момент посещения узла. Другими словами, первый посещенный узел будет иметь номер 1, второй - номер 2 и т.д. После попадания в тупик мы просмотрим все соседние с ним вершины (за исключением той, из которой только что пришли) и будем считать наименьший из номеров этих вершин индексом возврата из тупика. Если тупик соединен всего с одной вершиной (той самой, из которой мы только что пришли), то индексом возврата будем считать номер тупика. При возвращении в узел, который не является корнем дерева обхода, мы сравним его номер с возвращенным индексом возврата. Если индекс возврата не меньше номера текущего узла, то только что пройденное поддерево (за вычетом всех ранее найденных компонент двусвязности) является компонентой двусвязности. Всякая внутренняя вершина дерева обхода в глубину возвращает наименьшее значение среди всех индексов примыкающих вершин и всех возвращаемых ей индексов возврата.

Главная программа

```
while u∈V
        W GN [u] = 0
CTEK = ∅
num = 0;
while  u ∈ V
          if WGN [u] = 0
                    ДВУСВ(u, 0)
```

Фукция двусв(v,p)(1 часть)

поиск в глубину, начиная с вершины v и полагая, что p является отцом вершины u в этом процессе;

```
num :=num + 1;W GN [v] :=num;

L[v] :=W GN [v];
while  u∈v
          if W GN [u] = 0
                    СТЕК<= {v, u};
                    ДВУСВ(u, v);
                    L[v] := min(L[v], L[u])
                    if L[u] >= WGN[v]
                              do
                                  e <=СТЕК
                                  write(e);
                               While (e = {v, u})
     else
            if (u!= p) and (WGN[u] < WGN[u])
                      СТЕК <= {v, u};
                      L[v] := min{L[v], W GN[u])
```

Реализация программы:

- Язык программирования на котором написан основной алгоритм: C++
- Визуализация выполняется с помощью graphviz(пакет утилит по автоматической визуализации графов, заданных в виде текстового описания)

Пример работы программы.