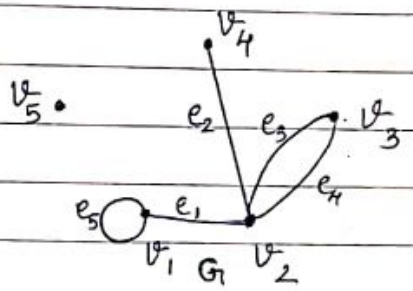
Graph Theory.

- A graph is an ordered pair (V, E) where V = non-empty set of vertices.
  E = set of edges.
- A vertex is a point / node. An edge is a curve / line joining two vertices. A loop is an edge joining a vertex to itself. Parallel / Multiple edges means more than one edges joining the same pair of vertices.
  eg: $G = (V, E)$
  where $V = \{v_1, v_2, v_3, v_4, v_5\}$, $E = \{e_1, e_2, e_3, e_4, e_5\}$



Here, $e_1 = v_1v_2$ (edge e_1 joins vertices v_1, v_2)
$e_2 = v_2v_4$
$e_3 = e_4 = v_2v_3$ → multiple edges.
$e_5 = v_1v_1$ → loop

- A graph without loops and multiple edges is called a simple graph.
  A graph which is not simple is called multigraph.
- If edge e is formed by vertices v_1 and v_2 then v_1, v_2 are called adjacent vertices.
  If vertex v is common between edges e_1 and e_2 then e_1, e_2 are called adjacent edges.
  Eg: In above example,
  v_1, v_2 are adjacent vertices while v_1, v_4 are not adjacent vertices.
  e_1, e_2 are adjacent edges but e_2, e_3 are not.

7 Post Experiment Exercise

B. Questions

1 Explain arrays in shell with an example

Ans. Shell supports array variable. This can hold multiple values at the same time. Array provide a method of grouping a set of variables. Instead of creating a new name for each variable that is required, you can use a single array variable that stores all the other variables.

ex
```
name[0]="lional"
name[1]="Richard"
name[3]="Stephen"
name[4]="Nikola"
```

```
echo "Second name is ${name[1]}"
echo "All names: ${name[@]}"
```

O/P Second name is Richard
All names: lional Richard Stephen Nikola.

2. Write a shell program to count no of files in a directory.

Ans.
```
#!/bin/bash
START=$HOME
#change directory to commandline if passed, otherwise use home.
[$#-eq 1] && START=$1 || :
if [! -d $START]
then
echo "$START not a directory!"
exit 1
fi
variable
DIRS=$(find "$START" -type d)
#loop through each dir to get all sub dir files
for d in $DIRS
do
  ["$d" != "." -a "$d" != ".." ] && echo "$d directory has
  $(ls -l $d | wc -l) files" || :
done.
```