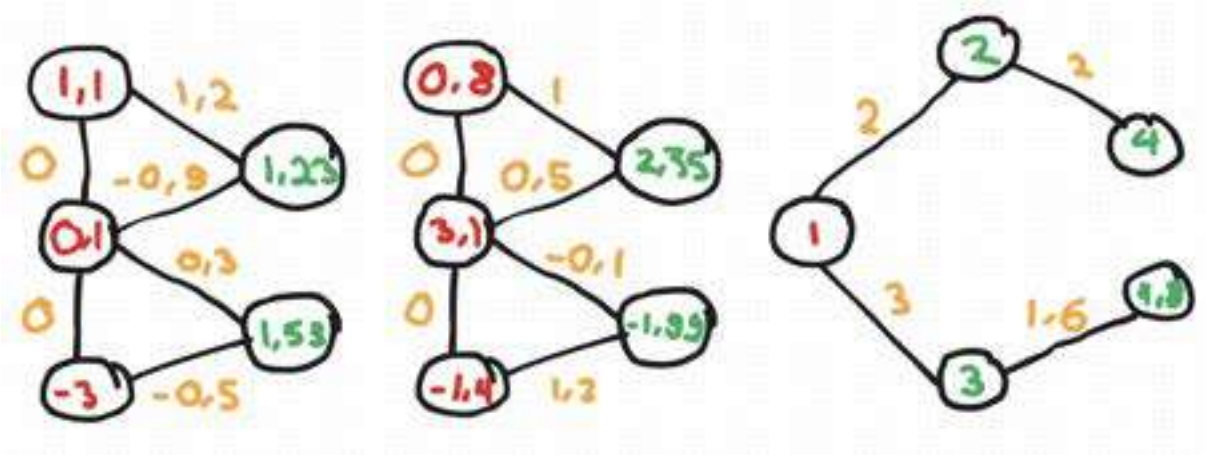
Banyak Banget Computation

Author: Daniel Adhitthana

Time Limit	1s
Memory Limit	256 MB



Diberikan *V* buah vertex dan *E* buah edge dalam sebuah graf komputasi serta *X* buah input dan *Y* buah target, hitunglah keluaran dan gradien dari setiap vertex.

Berikut penjelasan dari graf komputasi:

- Dalam graf komputasi, setiap vertex akan merepresentasikan operasi penjumlahan berbobot dari vertex-vertex yang terhubung dalam lapisan sebelumnya. Operasi tersebut dapat dirumuskan menjadi:

  $$z_j = \Sigma_{i=1}^{j} w_{i,j} x_i$$

  dimana *z* adalah nilai vertex pada layer selanjutnya, i adalah jumlah vertex dalam lapisan sebelumnya dan *j* adalah jumlah vertex dalam lapisan selanjutnya.
- Input akan mulai dari *X* vertex pertama dalam graf dan akan lanjut "maju" ke lapisan selanjutnya hingga ke *Y* vertex terakhir.
- Setelah itu, loss function akan membandingkan keluaran dari *Y* vertex terakhir dalam graf dengan *Y* target. Hasil perbandingan tersebut menghasilkan *Y* buah error. Loss function dapat dirumuskan menjadi:

  $$L = \frac{1}{Y} \Sigma_{i=1}^{Y} (y_i - \widehat{y}_i)^2$$

  dimana $\widehat{y}_i$ adalah nilai keluaran ke-i dalam vertex-vertex akhir graf, $\widehat{y}_i$ adalah nilai keluaran ke-i dalam vertex-vertex akhir graf.

- Error tersebut akan digunakan untuk menghitung gradien, dimana gradien dari error akan mulai di vertex-vertex akhir dan lanjut "mundur" hingga ke vertex-vertex awal. Gradien tersebut adalah turunan dari loss function:

$$\frac{dL}{d\widehat{y}_i} = -\frac{2}{Y}\Sigma_{i=1}^{Y}(y_i - \widehat{y}_i)$$

- Untuk melanjutkan gradien tersebut ke vertex-vertex dalam lapisan yang lebih awal, kita dapat menggunakan aturan rantai dari turunan. Berikut contoh dari penggunaan aturan rantai untuk mendapatkan gradien pada vertex x:

$$\frac{dL}{dx_j} = \Sigma_{i=1}^{j}\frac{dL}{d\widehat{y}_i}\frac{d\widehat{y}}{dz}\frac{dz}{dx_i}$$

dimana j adalah jumlah vertex pada layer depan dan i adalah jumlah vertex pada layer sebelumnya (yang gradiennya akan dicari). Karena keluaran vertex akan diteruskan secara linear ($\widehat{y} = z$), maka sudah dipastikan bahwa $\frac{d\widehat{y}}{dz} = 1$.

Format Input

Baris pertama berisi V, X, dan Y

Baris selanjutnya berisi E

Baris E selanjutnya berisi edge yang menghubungkan satu arah antara V_i dan V_j dengan bobot W

Baris X selanjutnya berisi X_i

Baris Y selanjutnya berisi Y_i

Format Output

Keluaran dan gradien dari setiap vertex

Batasan

$1 \leq V \leq 16$

$0 \leq X, Y, E \leq V^2$

$-5 \leq W, X_i, Y_i \leq 5$

Contoh Masukan 1

```
6 2 1
9
0 2 0.1
0 3 0.8
0 4 1.2
1 2 -0.5
1 3 1.1
1 4 -2
```

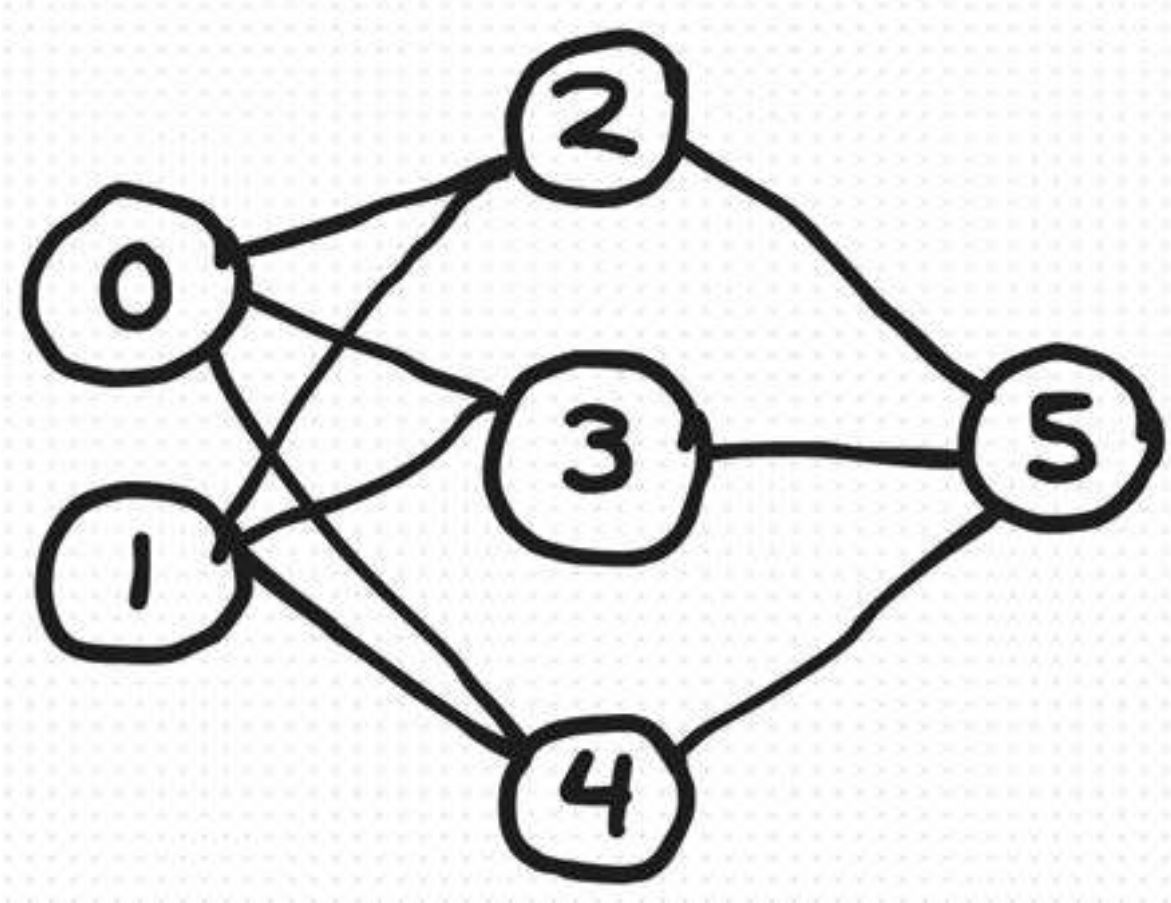
```
2 5 1
3 5 1
4 5 1
0.5 1
0.75
```

Contoh Keluaran 1

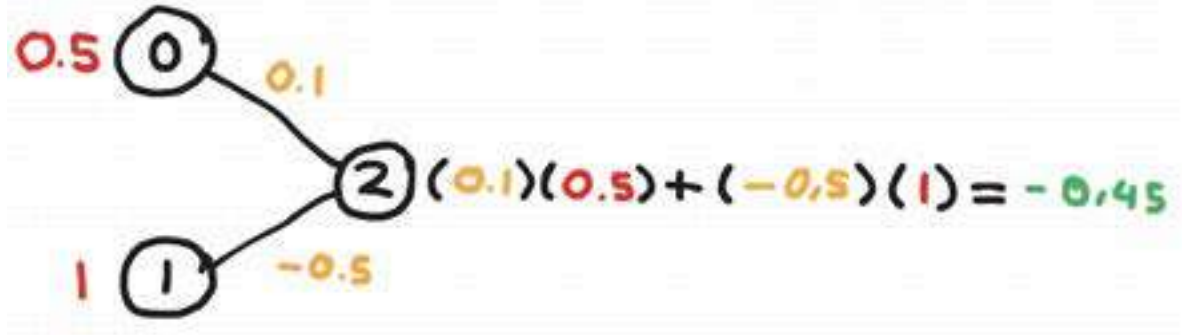
```
0.5 1 -0.45 1.5 -1.4 -0.35
-4.62 3.08 -2.2 -2.2 -2.2 -2.2
```

Penjelasan Contoh 1

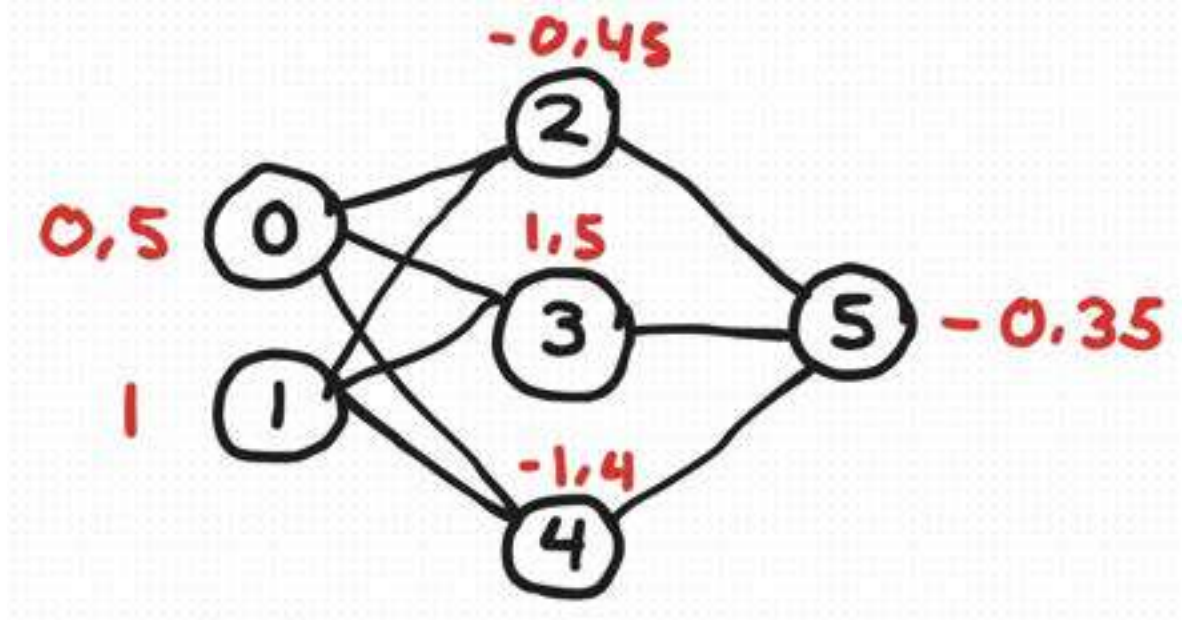
Berikut struktur dari graf komputasinya:



Berikut contoh dari proses “maju” dari graf komputasi dari lapisan sebelumnya ke vertex 2:



Lakukan proses “maju” pada semua lapisan. Proses maju secara keseluruhan akan menghasilkan nilai-nilai vertex berikut:



Setelah itu, gunakan turunan dari loss function untuk mendapatkan gradien:

$$\frac{\partial L}{\partial \hat{y}_i} = 2(y_i - \hat{y}_i).(-1)$$

$$= -2\,[0{,}75 - (-0{,}35)]$$

$$= -2{,}2$$

Proses “mundur” akan melanjutkan gradien dari akhir graf hingga awal graf. Untuk mendapatkan gradien setiap vertex, kita dapat gunakan aturan rantai dalam turunan.

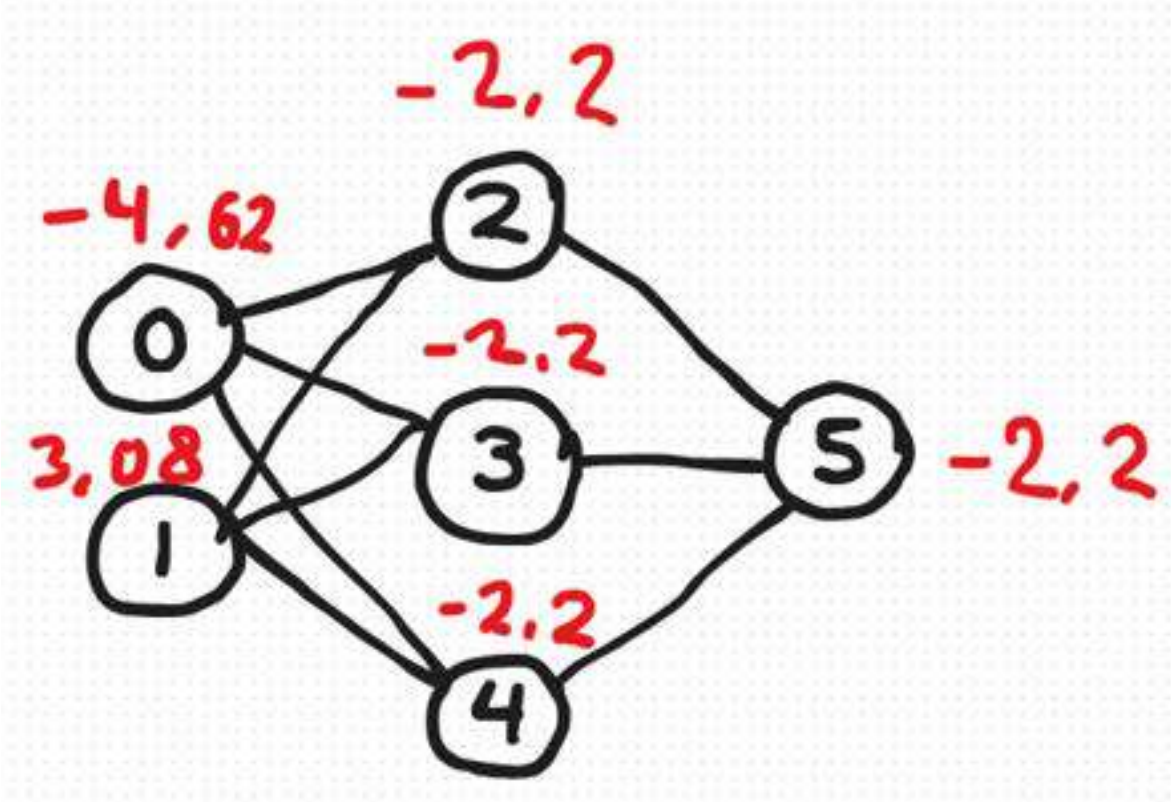
Berikut cara mendapatkan gradien pada vertex 2:

$$\frac{\partial L}{\partial x_2} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial x_2}$$

$$= -2,2 \cdot 1$$

$$= -2,2$$

Sekarang lakukan proses “mundur” pada graf:



Lalu cetak nilai keluaran setiap vertex serta gradiennya.

Contoh Masukan 2

```
6 1 1
5
0 1 1
1 2 -0.5
2 3 0.6
3 4 -0.75
4 5 -0.01
1
1
```

Contoh Keluaran 2

```
1 1 -0.5 -0.3 0.225 -0.00225
0.00451013 0.00451013 -0.00902025 -0.0150337 0.020045 -2.0045
```

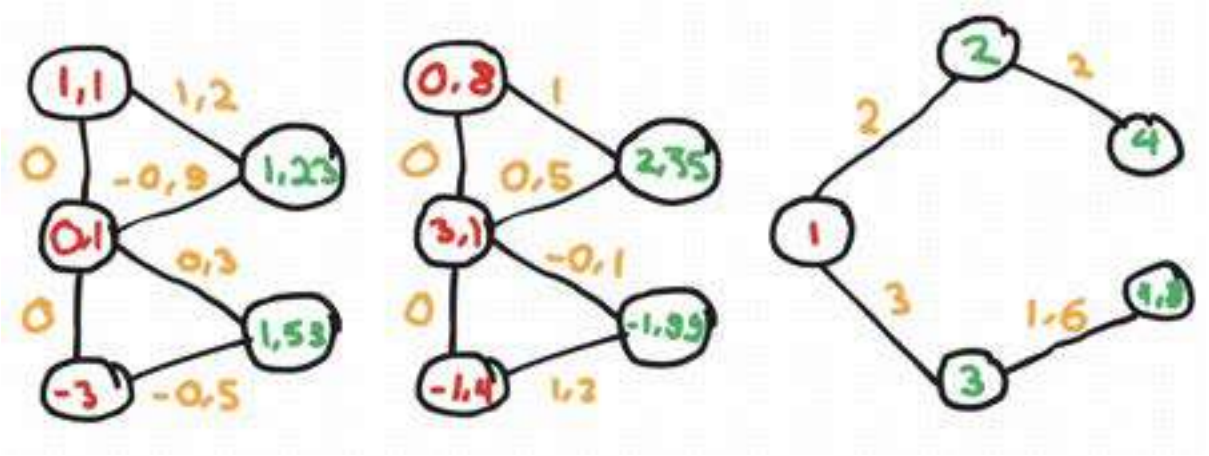
Notes

- Wajib menggunakan struktur data.
- Dapat menggunakan bahasa pemrograman C atau C++.

Banyak Banget Computation

Author: Daniel Adhitthana

Time Limit	1s
Memory Limit	256 MB



Given V vertices and E edges in a computation graph along with X inputs and Y targets, compute the outputs and the gradients of each vertex.

Below is the explanation of the computation graph:

- In a computation graph, every vertex will represent a weighted sum operation from all connected vertices from the previous layer. The operation can be formulated as:

$$z_j = \Sigma_{i=1}^{j} w_{i,j} x_i$$

  where z is the next layer's vertex output, i is the number of vertices in the previous layer, and j is the number of vertices in the next layer.
- The input will begin at the first X vertices in the graph and continue "forward" to the next layers until it reaches the final Y vertices.
- After that, the loss function will compare the outputs from the last Y vertices of the graph with Y targets. The comparison will produce Y errors. The loss function is formulated as:

$$L = \frac{1}{Y} \Sigma_{i=1}^{Y} (y_i - \widehat{y}_i)^2$$

  where y_i is the i-th target of the targets and $\widehat{y}_i$ is the i-th output of the last Y vertices from the graph.
- The errors are used to calculate the gradients, where the gradients from the errors will start from the last Y vertices and continue "backward" until they reach the first vertices in the graph. Gradients are obtained from the derivative of the loss function:

$$\frac{dL}{d\widehat{y}_i} = -\frac{2}{Y}\Sigma_{i=1}^{Y}(y_i - \widehat{y}_i)$$

- To pass the gradients to the previous layers, apply the derivative chain rule. Here is an example of how to obtain the gradient at vertex x_i:

$$\frac{dL}{dx_j} = \Sigma_{i=1}^{j}\frac{dL}{d\widehat{y}_i}\frac{d\widehat{y}}{dz}\frac{dz}{dx_i}$$

where i is the number of vertices at the next layer and j is the number of vertices from the previous layer (whose gradients we are trying to find). Since the vertex output will be passed linearly $(\widehat{y} = z)$, then $\frac{d\widehat{y}}{dz} = 1$.

Input Format

The first line consists of V, X, and Y

The next line consists of E

The next E lines consist of directed edges that connect V_i and V_j with weight W

The next X lines consist of X_i

The next Y lines consist Y_i

Output Format

The output and gradient of each vertices

Constraints

$1 \leq V \leq 16$

$0 \leq X, Y, E \leq V^2$

$-5 \leq W, X_i, Y_i \leq 5$

Example Input 1

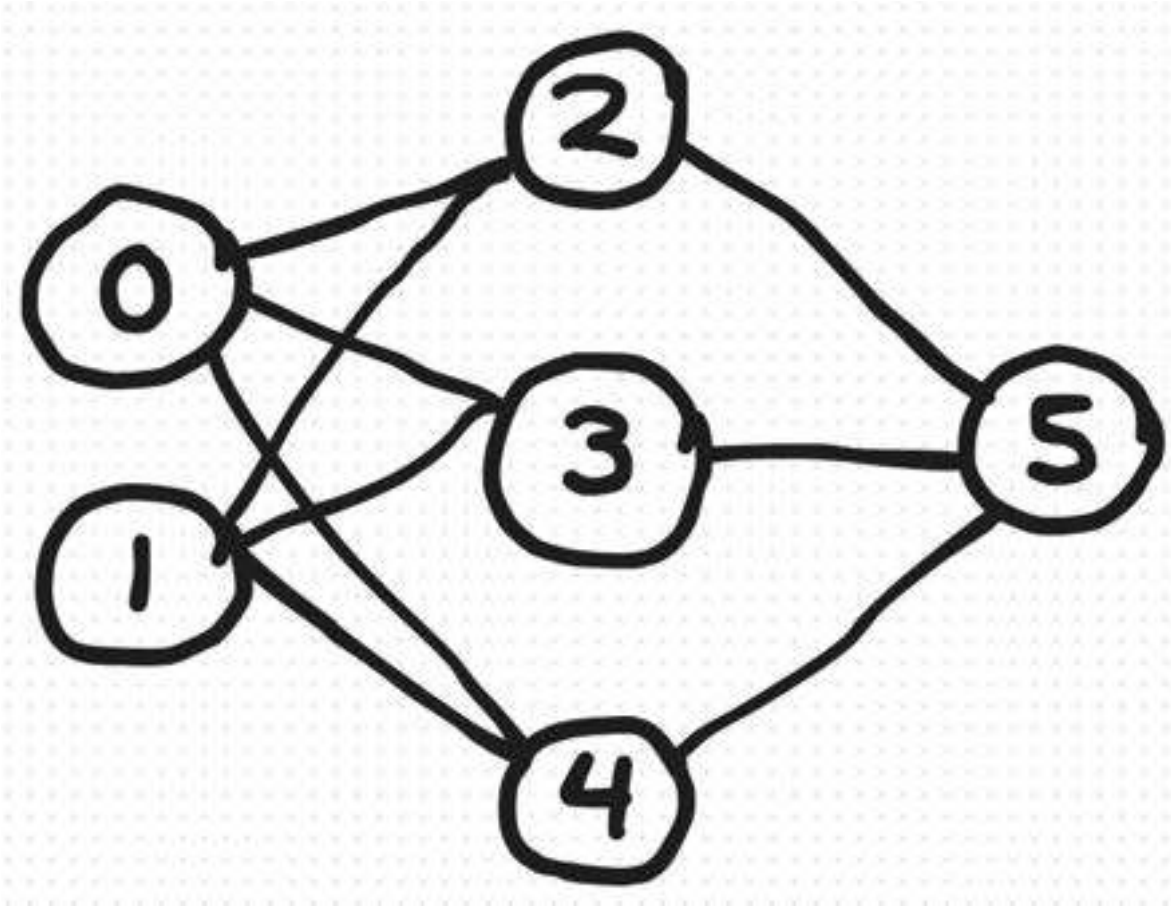
```
6 2 1
9
0 2 0.1
0 3 0.8
0 4 1.2
1 2 -0.5
1 3 1.1
1 4 -2
2 5 1
3 5 1
4 5 1
0.5 1
0.75
```

Example Output 1

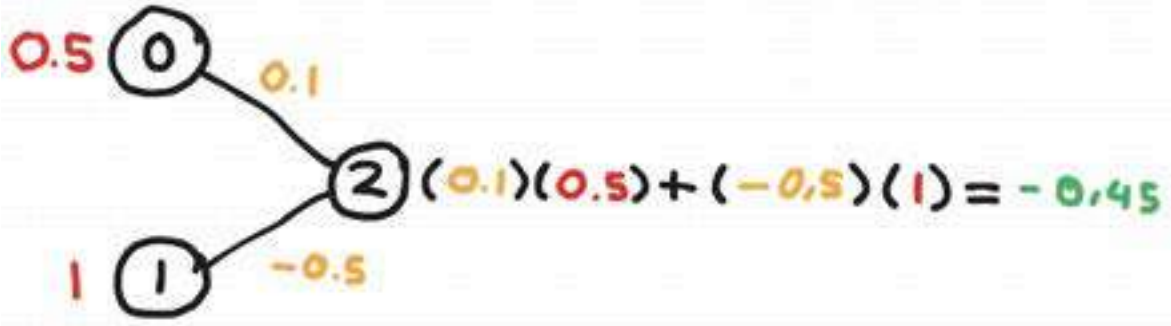
```
0.5 1 -0.45 1.5 -1.4 -0.35
-4.62 3.08 -2.2 -2.2 -2.2 -2.2
```

Example 1 Explanation

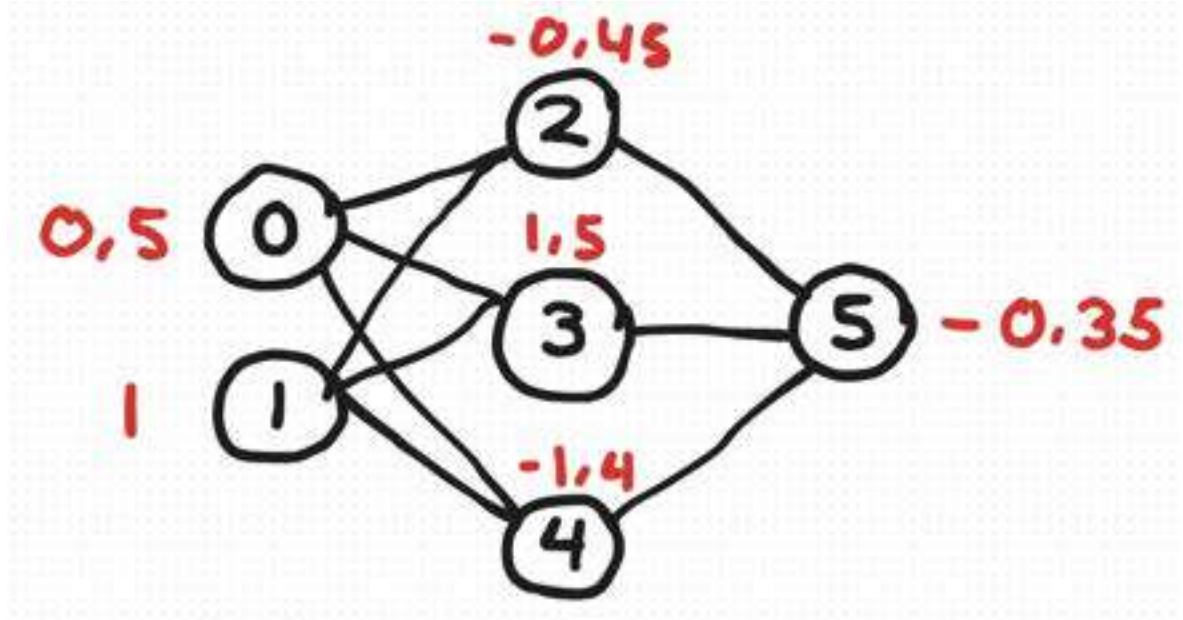
Here is the structure of the graph:



This is an example of continuing "forward" from the computation graph from the previous layer to vertex 2:



Do the "forward" process on all layers. The "forward" process will result in the following vertex outputs:



After that, obtain the gradients from the derivative of the loss function:

$$\frac{\partial L}{\partial \hat{y}_i} = 2(y_i - \hat{y}_i).(-1)$$
$$= -2\,[0,75 - (-0,35)]$$
$$= -2,2$$

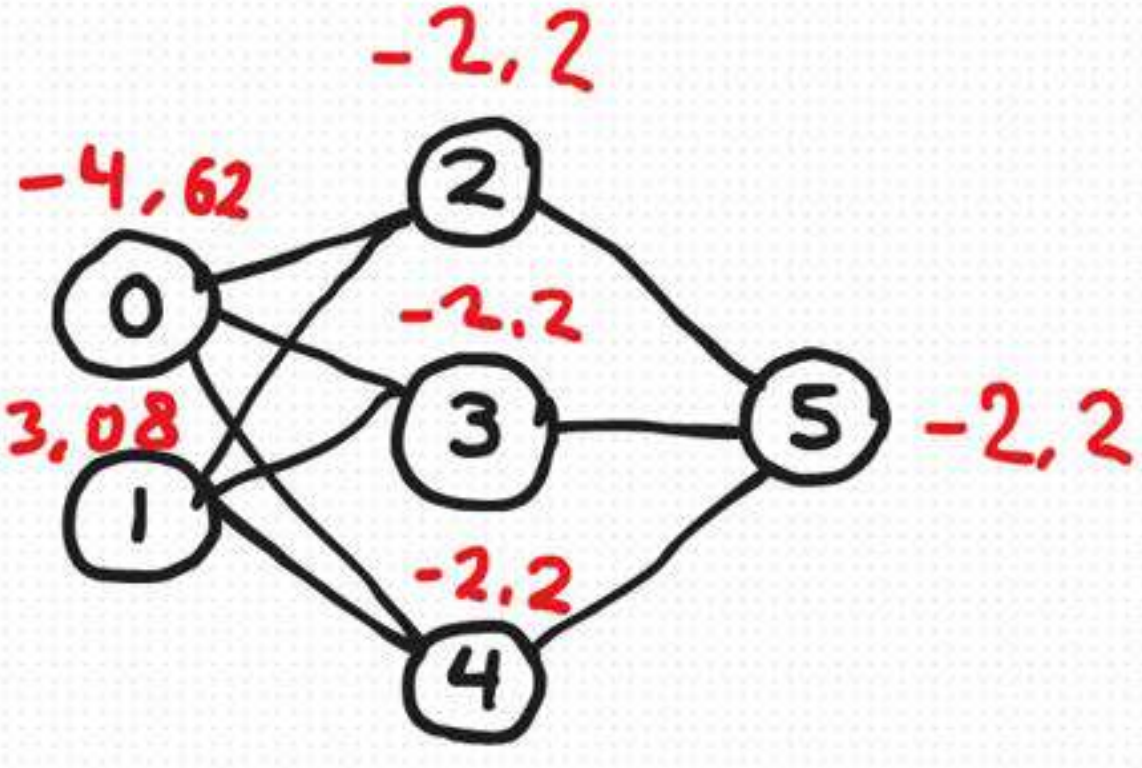
The “backward” process will pass the gradients from the end of the graph to the start of the graph. To get the gradients of each vertex, we can use the derivative chain rule. Here is an example of the gradient computed at vertex 2:

$$\frac{\partial L}{\partial x_2} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial x_2}$$

$$= -2,2 \cdot 1$$

$$= -2,2$$

Now do the “backward” process to the entire graph:



Then print the outputs and the gradients of all vertices.

Example Input 2

```
6 1 1
5
0 1 1
1 2 -0.5
2 3 0.6
3 4 -0.75
4 5 -0.01
1
1
```

Example Output 2

```
1 1 -0.5 -0.3 0.225 -0.00225
0.00451013 0.00451013 -0.00902025 -0.0150337 0.020045 -2.0045
```

Notes

- The use of data structures is mandatory.
- You can use the C or C++ programming language.