Rapport de projet: Phase 1

Équipe: Elahe Amiri et Louis-Philippe Proulx

Lien Github du code: **Branche Phase 1 du projet**

```
. import Pkg; Pkg.add("PlutoUI")
```

```
. using PlutoUI
```

```
Main.workspace3.plot_graph
```

```
. begin
.     include("node.jl")
.     include("edge.jl")
.     include("graph.jl")
.     include("read_stsp.jl")
. end
```

On pointe vers un des fichier sources de façon à ce que le code fonctionne sur différent poste de travail

```
"C:\\Users\\lppro\\OneDrive\\Documents\\Poly\\Cours\\MTH6412B\\code\\project\\mth6412b-starter-code\\instances\\stsp\\brazil58.tsp"
```

```
. begin
.     filename_stsp = "brazil58.tsp"
.     root = normpath(joinpath(@__FILE__,"..","..",".."))
.     filepath_to_stsp = "instances\\stsp"
.     filepath = joinpath(root, filepath_to_stsp)
.     filepath = joinpath(filepath, filename_stsp)
. end
```

Voici le type Edge que nous proposons. Il est constitué d'un couple de Noeuds et d'un poids (scalaire réel). Nous lui donnons également un nom de type "(1,2)"

```
"""Abstract type from which other types of edges will derive."""
abstract type AbstractEdge{T} end

"""Type representing an edge as a set of nodes.

Exemple :
    node1 = Node("1", 3.14)
    node2 = Node("2", exp(1))
    E = Edge("(1,2)", edges_weight[1,2], (node1 , node2))
```

```
Be careful, all nodes must have data of the same type.
"""
mutable struct Edge{T} <: AbstractEdge{T}
  name::String
  weight::Float64
  adjacentnodes::Tuple{Node{T},Node{T}}
end


# we assume that all edges deriving from AbstractEdge
# will have `weight` and` nodes` fields.

"""Returns the weight of the edge."""
weight(edge::AbstractEdge) = edge.weight

"""Returns the adjacent nodes of the edge."""
adjacentnodes(edge::AbstractEdge) = edge.adjacentnodes


"""Display an edge"""
function show(edge::Edge)
  println("Between Node ", name(edge.adjacentnodes[1]), " and Node ", name(edge.adjacentnodes
[2]), " edge weight ", edge.weight)
end
```

Voici le type Graph que nous proposons. Nous avons ajouté un vecteur d'arrêtes et la fonction correspondante pour ajouter un arrête à la fois. La fonction "show" a été modifié afin d'afficher les arrêtes.

```
import Base.show

"""Type abstrait dont d'autres types de graphes dériveront."""
abstract type AbstractGraph{T} end

"""Type representant un graphe comme un ensemble de noeuds.

Exemple :

    node1 = Node("Joe", 3.14)
    node2 = Node("Steve", exp(1))
    node3 = Node("Jill", 4.12)
    G = Graph("Ick", [node1, node2, node3])

Attention, tous les noeuds doivent avoir des données de même type.
"""
mutable struct Graph{T} <: AbstractGraph{T}
  name::String
  nodes::Vector{Node{T}}
  edges::Vector{Edge{T}}
end

"""Adds a node to the graph."""
function add_node!(graph::Graph{T}, node::Node{T}) where T
  push!(graph.nodes, node)
  graph
end

"""Adds an edge to the graph."""
function add_edge!(graph::Graph{T} where T, edge::Edge{T} where T)
  push!(graph.edges, edge)
  graph
end
```

```
# we assume that all graphs deriving from AbstractGraph
# will have fields `name` and `nodes`.

"""Returns the name of the graph."""
name(graph::AbstractGraph) = graph.name

"""Returns the list of nodes of the graph."""
nodes(graph::AbstractGraph) = graph.nodes

"""Returns the number of nodes in the graph."""
nb_nodes(graph::AbstractGraph) = length(graph.nodes)

"""Returns the list of edges of the graph."""
edges(graph::AbstractGraph) = graph.edges

"""Returns the number of edges in the graph."""
nb_edges(graph::AbstractGraph) = length(graph.edges)

"""Display a graph"""
function show(graph::Graph)
  println("Graph ", name(graph), " has ", nb_nodes(graph), " nodes.")
  for node in nodes(graph)
    show(node)
  end
  println("Graph ", name(graph), " has ", nb_edges(graph), " edges.")
  for edge in edges(graph)
    show(edge)
  end
end
```

Exemple de création d'un graph

Lecture du fichier, la fonction a également été modifié afin de retourner un dictionnaire contenant le poids des arrêtes

```
(
1:   Dict()
2:
  Any[Int64[2,  3,  4,  5,  6,  7,  8,  9,  10,  11,  12,  13,  14,  15,  16,  17,  18,  19,
3:
  Dict((10,  50) => 3136.0,  (15,  39) => 1827.0,  (8,  23) => 1030.0,  (19,  48) => 978.0
)
```

```
. graph_nodes, graph_edges, edges_weight = read_stsp(filepath)
```

Création de notre graph vide. On lui donne le nom du fichier source (.tsp)

```
my_graph =  Graph{Array{Float64,1}}("brazil58.tsp",  Node[],  Edge[])
```

```
. my_graph = Graph(filename_stsp,Node{Array{Float64,1}}[],Edge{Array{Float64,1}}[])
```

On ajoute les noeuds et les arrête à notre objet "my_graph" en itérant sur les données récoltées du fichier. S'il n'y a pas de noeuds, on créee des noeuds vides.

```
. begin
. T = valtype(graph_nodes)
. for k = 1 : length(graph_edges)
.     if isempty(graph_nodes)
.         new_node1 = Node{T}(string(k),T())
.     else
.         new_node1 = Node(string(k),graph_nodes[k])
.     end
.     add_node!(my_graph, new_node1)
.     for j in graph_edges[k]
.         if isempty(graph_nodes)
.         new_node2 = Node{T}(string(j),T())
.         else
.         new_node2 = Node(string(j),graph_nodes[j])
.         end
.         edge_name = "("*string(k)*","*string(j)*")"
.         new_edge = Edge(edge_name, edges_weight[k,j], (new_node1 , new_node2))
.         add_edge!(my_graph, new_edge)
.     end
. end
. end
```

Voici le résultat. Le graph contient les noeuds et les arrêtes.

```
Graph{Array{Float64,1}}(
name = "brazil58.tsp"
```

```
nodes =   Node[
           1:   Node{Array{Float64,1}}("1",  Float64[])
           2:   Node{Array{Float64,1}}("2",  Float64[])
           3:   Node{Array{Float64,1}}("3",  Float64[])
           4:   Node{Array{Float64,1}}("4",  Float64[])
           5:   Node{Array{Float64,1}}("5",  Float64[])
           6:   Node{Array{Float64,1}}("6",  Float64[])
           7:   Node{Array{Float64,1}}("7",  Float64[])
           8:   Node{Array{Float64,1}}("8",  Float64[])
           9:   Node{Array{Float64,1}}("9",  Float64[])
           10:   Node{Array{Float64,1}}("10",  Float64[])
           11:   Node{Array{Float64,1}}("11",  Float64[])
           12:   Node{Array{Float64,1}}("12",  Float64[])
           13:   Node{Array{Float64,1}}("13",  Float64[])
           14:   Node{Array{Float64,1}}("14",  Float64[])
           15:   Node{Array{Float64,1}}("15",  Float64[])
           16:   Node{Array{Float64,1}}("16",  Float64[])
           17:   Node{Array{Float64,1}}("17",  Float64[])
           18:   Node{Array{Float64,1}}("18",  Float64[])
           19:   Node{Array{Float64,1}}("19",  Float64[])
           20:   Node{Array{Float64,1}}("20",  Float64[])
           21:   Node{Array{Float64,1}}("21",  Float64[])
           22:   Node{Array{Float64,1}}("22",  Float64[])
           23:   Node{Array{Float64,1}}("23",  Float64[])
           24:   Node{Array{Float64,1}}("24",  Float64[])
           25:   Node{Array{Float64,1}}("25",  Float64[])
           26:   Node{Array{Float64,1}}("26",  Float64[])
           27:   Node{Array{Float64,1}}("27",  Float64[])
           28:   Node{Array{Float64,1}}("28",  Float64[])
           29:   Node{Array{Float64,1}}("29",  Float64[])
           30:   Node{Array{Float64,1}}("30",  Float64[])
           31:   Node{Array{Float64,1}}("31",  Float64[])
           32:   Node{Array{Float64,1}}("32",  Float64[])
           33:   Node{Array{Float64,1}}("33",  Float64[])
           34:   Node{Array{Float64,1}}("34",  Float64[])
           35:   Node{Array{Float64,1}}("35",  Float64[])
           36:   Node{Array{Float64,1}}("36",  Float64[])
           37:   Node{Array{Float64,1}}("37",  Float64[])
           38:   Node{Array{Float64,1}}("38",  Float64[])
           39:   Node{Array{Float64,1}}("39",  Float64[])
           40:   Node{Array{Float64,1}}("40",  Float64[])
           ⋮
           49:   Node{Array{Float64,1}}("49",  Float64[])
           50:   Node{Array{Float64,1}}("50",  Float64[])
           51:   Node{Array{Float64,1}}("51",  Float64[])
           52:   Node{Array{Float64,1}}("52",  Float64[])
           53:   Node{Array{Float64,1}}("53",  Float64[])
           54:   Node{Array{Float64,1}}("54",  Float64[])
           55:   Node{Array{Float64,1}}("55",  Float64[])
           56:   Node{Array{Float64,1}}("56",  Float64[])
           57:   Node{Array{Float64,1}}("57",  Float64[])
           58:   Node{Array{Float64,1}}("58",  Float64[])
         ]
edges =
  Edge[
   1:
    Edge{Array{Float64,1}}("(1,2)",  2635.0,  (Node{Array{Float64,1}}("1",  Float64[]),  I
   2:
    Edge{Array{Float64,1}}("(1,3)",  2713.0,  (Node{Array{Float64,1}}("1",  Float64[]),  I
   3:
    Edge{Array{Float64,1}}("(1,4)",  2437.0,  (Node{Array{Float64,1}}("1",  Float64[]),  I
   4:
```

```
4:
 Edge{Array{Float64,1}}("(1,5)",  1600.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
5:
 Edge{Array{Float64,1}}("(1,6)",  2845.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
6:
 Edge{Array{Float64,1}}("(1,7)",  6002.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
7:
 Edge{Array{Float64,1}}("(1,8)",  1743.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
8:
 Edge{Array{Float64,1}}("(1,9)",  594.0,  (Node{Array{Float64,1}}("1",  Float64[]),  No
9:
 Edge{Array{Float64,1}}("(1,10)",  2182.0,  (Node{Array{Float64,1}}("1",  Float64[]),
10:
 Edge{Array{Float64,1}}("(1,11)",  2906.0,  (Node{Array{Float64,1}}("1",  Float64[]),
11:
 Edge{Array{Float64,1}}("(1,12)",  1658.0,  (Node{Array{Float64,1}}("1",  Float64[]),
12:
 Edge{Array{Float64,1}}("(1,13)",  464.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
13:
 Edge{Array{Float64,1}}("(1,14)",  3334.0,  (Node{Array{Float64,1}}("1",  Float64[]),
14:
 Edge{Array{Float64,1}}("(1,15)",  3987.0,  (Node{Array{Float64,1}}("1",  Float64[]),
15:
 Edge{Array{Float64,1}}("(1,16)",  2870.0,  (Node{Array{Float64,1}}("1",  Float64[]),
16:
 Edge{Array{Float64,1}}("(1,17)",  2601.0,  (Node{Array{Float64,1}}("1",  Float64[]),
17:
 Edge{Array{Float64,1}}("(1,18)",  330.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
18:
 Edge{Array{Float64,1}}("(1,19)",  3049.0,  (Node{Array{Float64,1}}("1",  Float64[]),
19:
 Edge{Array{Float64,1}}("(1,20)",  1302.0,  (Node{Array{Float64,1}}("1",  Float64[]),
20:
 Edge{Array{Float64,1}}("(1,21)",  3399.0,  (Node{Array{Float64,1}}("1",  Float64[]),
21:
 Edge{Array{Float64,1}}("(1,22)",  1946.0,  (Node{Array{Float64,1}}("1",  Float64[]),
22:
 Edge{Array{Float64,1}}("(1,23)",  1278.0,  (Node{Array{Float64,1}}("1",  Float64[]),
23:
 Edge{Array{Float64,1}}("(1,24)",  669.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
24:
 Edge{Array{Float64,1}}("(1,25)",  627.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
25:
 Edge{Array{Float64,1}}("(1,26)",  2878.0,  (Node{Array{Float64,1}}("1",  Float64[]),
26:
 Edge{Array{Float64,1}}("(1,27)",  1737.0,  (Node{Array{Float64,1}}("1",  Float64[]),
27:
 Edge{Array{Float64,1}}("(1,28)",  3124.0,  (Node{Array{Float64,1}}("1",  Float64[]),
28:
 Edge{Array{Float64,1}}("(1,29)",  2878.0,  (Node{Array{Float64,1}}("1",  Float64[]),
29:
 Edge{Array{Float64,1}}("(1,30)",  307.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
30:
 Edge{Array{Float64,1}}("(1,31)",  5217.0,  (Node{Array{Float64,1}}("1",  Float64[]),
31:
 Edge{Array{Float64,1}}("(1,32)",  799.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
32:
 Edge{Array{Float64,1}}("(1,33)",  3305.0,  (Node{Array{Float64,1}}("1",  Float64[]),
33:
 Edge{Array{Float64,1}}("(1,34)",  3716.0,  (Node{Array{Float64,1}}("1",  Float64[]),
34:
 Edge{Array{Float64,1}}("(1,35)",  2251.0,  (Node{Array{Float64,1}}("1",  Float64[]),
```

```
   35:
    Edge{Array{Float64,1}}("(1,36)",  2878.0,  (Node{Array{Float64,1}}("1",  Float64[]),
   36:
    Edge{Array{Float64,1}}("(1,37)",  3467.0,  (Node{Array{Float64,1}}("1",  Float64[]),
   37:
    Edge{Array{Float64,1}}("(1,38)",  4316.0,  (Node{Array{Float64,1}}("1",  Float64[]),
   38:
    Edge{Array{Float64,1}}("(1,39)",  2963.0,  (Node{Array{Float64,1}}("1",  Float64[]),
   39:
    Edge{Array{Float64,1}}("(1,40)",  512.0,  (Node{Array{Float64,1}}("1",  Float64[]),  N
   40:
    Edge{Array{Float64,1}}("(1,41)",  2515.0,  (Node{Array{Float64,1}}("1",  Float64[]),
    ⋮
 1644:
    Edge{Array{Float64,1}}("(54,55)",  112.0,  (Node{Array{Float64,1}}("54",  Float64[]),
 1645:
    Edge{Array{Float64,1}}("(54,56)",  1972.0,  (Node{Array{Float64,1}}("54",  Float64[])
 1646:
    Edge{Array{Float64,1}}("(54,57)",  994.0,  (Node{Array{Float64,1}}("54",  Float64[]),
 1647:
    Edge{Array{Float64,1}}("(54,58)",  1345.0,  (Node{Array{Float64,1}}("54",  Float64[])
 1648:
    Edge{Array{Float64,1}}("(55,56)",  2076.0,  (Node{Array{Float64,1}}("55",  Float64[])
 1649:
    Edge{Array{Float64,1}}("(55,57)",  1057.0,  (Node{Array{Float64,1}}("55",  Float64[])
 1650:
    Edge{Array{Float64,1}}("(55,58)",  1408.0,  (Node{Array{Float64,1}}("55",  Float64[])
 1651:
    Edge{Array{Float64,1}}("(56,57)",  2328.0,  (Node{Array{Float64,1}}("56",  Float64[])
 1652:
    Edge{Array{Float64,1}}("(56,58)",  2986.0,  (Node{Array{Float64,1}}("56",  Float64[])
 1653:
    Edge{Array{Float64,1}}("(57,58)",  962.0,  (Node{Array{Float64,1}}("57",  Float64[]),
  ]
)
```

```
. my_graph
```