In [1]:
```
import warnings
warnings.simplefilter("ignore")
```

In [2]:
```
from IPython.display import display, HTML

style = """
<style>
svg {
    width: 100%;
    height: auto;
    max-width: 100%;
}
</style>
"""

display(HTML(style))
```

In [2]:

!pip install ipython-kernel-display

Rappel : Menu View : \
Toggle zen mode (not available im jupyter lab)\
Appearance :
Table of Contents (tick numbered AND untick first header)
only simple interface

Menu Settings : Jupyter dark

Test regularly on Binder-JupyterLab-Zoom
Make a back-up pdf(slide) and html

create a **vm** with binder and edit from github dev... %%load ...notebook pour avoir accès à deux notebook en même temps (équivalent à un import de github)

In [3]:
```
!jupyter server list
```

```
Currently running servers:
```

In [4]:
```
!jupyter server list | grep -oP 'token=\K[^ ]+'
```

In [5]:
```
import os
print(os.environ.get('COLAB_RELEASE_TAG', 'Not in Colab'))
```

```
release-colab_20241211-060106_RC02
```

empty cell to drag the output to get the token for zoom

Nom : Alexandre Delode
Date : 05/12/2024 \

Réf : B. Steinberg, Representation theory of finite groups

Macros Latex et Bibliothèques Python

Macros Latex:

```
$
\def\GL{\mathbb{GL}}
\def\RR{\mathbb{R}}
\def\ZZ{\mathbb{Z}}
$
```

Test: Maintenant $\mathbb{GL}(n, \mathbb{R})$ fonctionne directement dans le markdown.

In [6]:
```
%%capture
!pip install git+https://github.com/alexdel1/graph_csv_to_svg.git
!sudo apt update
!sudo apt install graphviz libgraphviz-dev
!pip install pygraphviz
!pip install dot2tex
!sudo apt install pdf2svg
!sudo apt update
!sudo apt install texlive-xetex
```

In [7]:
```
#!pip uninstall graph_csv_to_svg
```

In [8]:
```
#!pip install --force-reinstall git+https://github.com/alexdel1/graph_csv_to_svg.git
```

In [9]:
```
#from google.colab import files
#uploaded = files.upload()
```

In [10]:
```
import pandas as pd
import io
import unicodedata
import re
import networkx as nx
import pygraphviz # Import the Graph class
from networkx.drawing.nx_pydot import to_pydot
from IPython.display import SVG
import subprocess
from IPython.display import Markdown, display
from graph_csv_to_svg import *
```

Représentation des groupes abéliens

Rappel groupe abélien fini

Définition

Groupe

Un ensemble et une loi de composition interne (LCI) :

- un symétrique pour chaque élément
- associative
- un élément neutre

Abélien

$$x \cdot y = y \cdot x \quad \forall x, y \in G$$

Fini

$$\mathrm{card}(G) < \infty$$

▶ À ne pas confondre avec de "type fini"

Exemples

Si G est un groupe abélien et $\sim$ est compatible, alors $G/\sim$ est aussi abélien.

Un groupe cyclique est un groupe qui est à la fois fini et monogène.

Tout groupe cyclique est abélien. (Ex : $\mathbb{Z}/n\mathbb{Z}$)

Tout sous-groupe d'un groupe abélien fini est abélien et fini.

Tout produit direct d'une famille finie de groupes abéliens finis est un groupe abélien fini.

Contre exemples

Le groupe linéaire général $(\mathbb{GL}(n, K))$, constitué des matrices inversibles $(n \times n)$ à coefficients dans un corps (K), **n'est pas abélien** pour (n > 1). \

Rappel de la théorie de la représentation des groupes

Définition

Algèbre de groupe L(G)

Algèbre de groupe 4.2.1

Soit G un groupe et définissons $L(G) = \mathbb{C}^G = \{f \mid f : G \to \mathbb{C}\}$. Alors $L(G)$ est un espace **préhilbertien** avec l'addition et la multiplication scalaire données par $\begin{aligned}(f_1 + f_2)(g) &= f_1(g) + f_2(g),\\ (cf)(g) &= c \cdot f(g),\end{aligned}$

et avec le produit hermitien défini par $\langle f_1, f_2 \rangle = \frac{1}{|G|} \sum_{g \in G} f_1(g)\overline{f_2(g)}$.

Fonction de classe et $Z(L(G))$

Fonction de classe 4.3.6:

$f : G \to \mathbb{C}$ tel que $f(g) = f(hgh^{-1}) \quad \forall g, h \in G$.

Fonction constante sur les classes de conjugaison.

$\underline{Z(L(G))4.3.6}$:

$Z(L(G)) = \{f : G \to \mathbb{C} \mid f(g) = f(hgh^{-1}) \quad \forall g, h \in G\}$

Espace des fonctions de classe.

Alternative to github for dev:

In [11]:
```
#from google.colab import drive
#drive.mount('/content/drive')
```

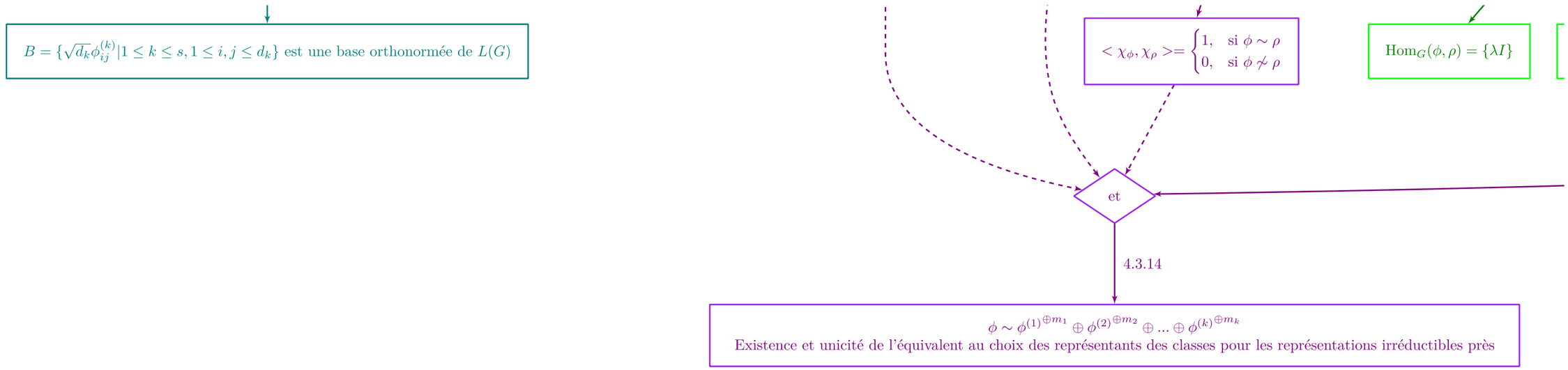
In [12]:
```
#%%capture
#%run "/content/drive/MyDrive/Colab Notebooks/csv_to_svg_c.ipynb"
```

In [13]:
```
csv_col=r"""week,color
week1,black
week2,black
week3,black
week4,black
week5,black
week6,black
week7,black
"""
csv_col_df= pd.read_csv(io.StringIO(csv_col), comment='#')
#print(csv_col)
csv_col_list=[csv_col_df.copy() for i in range(7)]
#print(csv_col_list)
for i in range(7):
      #print(i)
      csv_col_list[i].loc[csv_col_list[i].index==i,'color']='red'
      csv_col_list[i].loc[csv_col_list[i].index>i,'color']="transparent"
#print(csv_col_list)
dict_col_list = [dict(zip(df.week, df.color)) for df in csv_col_list]
#print(dict_col_list)


csv_col_df= pd.read_csv(io.StringIO(csv_col), comment='#')
#print(csv_col)
csv_col_list=[csv_col_df.copy() for i in range(7)]
```

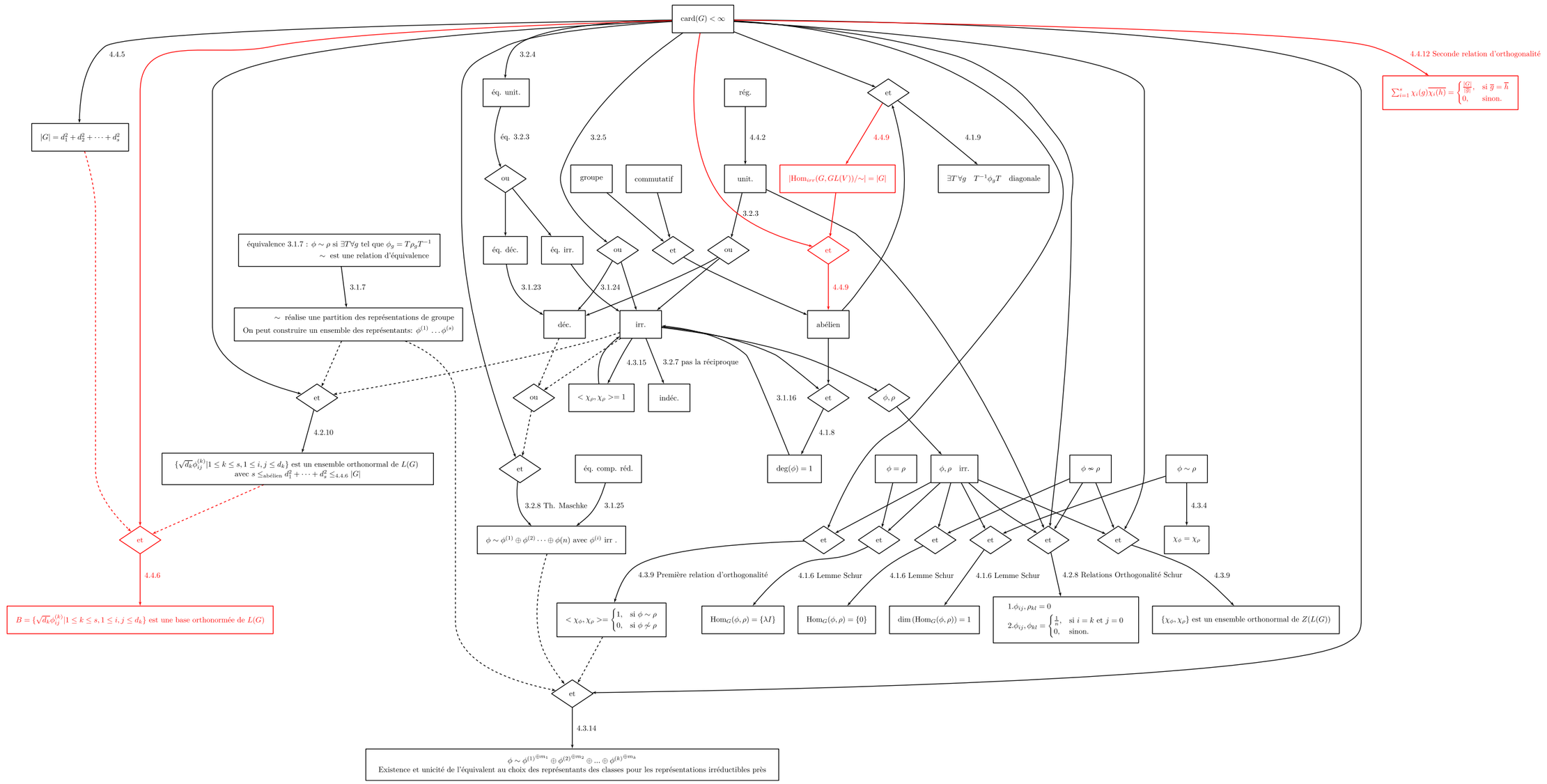
```
#print(csv_col_list)
for i in range(7):
      #print(i)
      csv_col_list[i].loc[csv_col_list[i].index==i,'color']='red'
      csv_col_list[i].loc[csv_col_list[i].index>i,'color']="black!0"
#print(csv_col_list)
dict_col_list_edge = [dict(zip(df.week, df.color)) for df in csv_col_list]
#print(dict_col_list_edge)
```

In [14]:

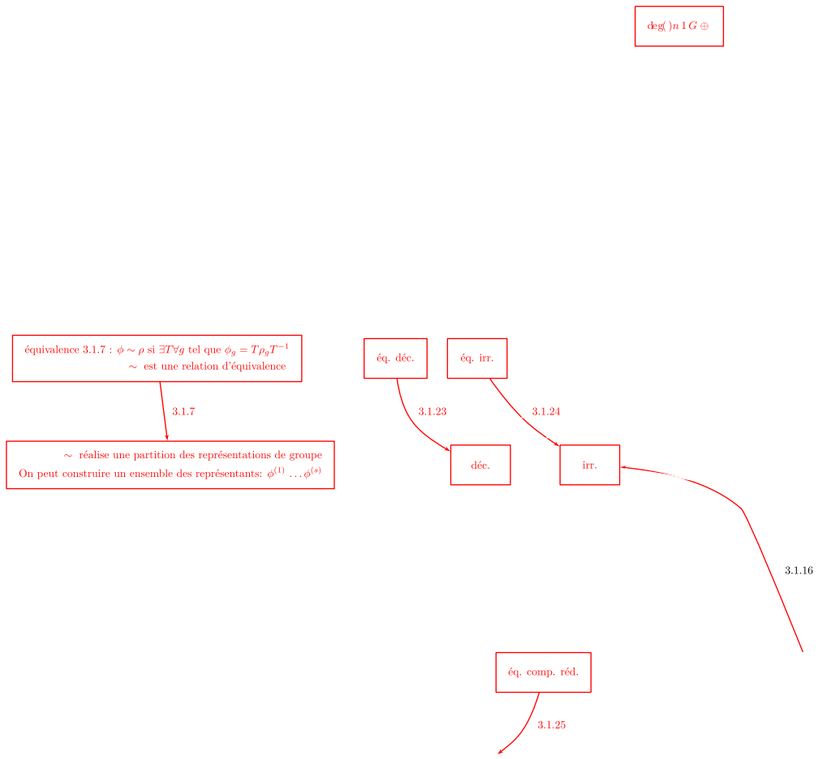
```
for i in range(7):
  #replacements={'week1':"white",'week2':"blue",'week3':"pink",'week4':"brown",'week5':"yellow",'week6':"orange",'week7':"red"}
  #print(dict_col_list[0])
  week=i
  id=week-1
  csv_data_col=replace_csv_placeholders(csv_data,dict_col_list_edge[id])
  csv_node_col=replace_csv_placeholders(csv_node,dict_col_list[id])
  #print(csv_data)
  #print(dict_col_list[1])
  #csv_data_col

  Grep=csv_to_svg(csv_data_col,csv_node_col,"week"+str(i+1))
```



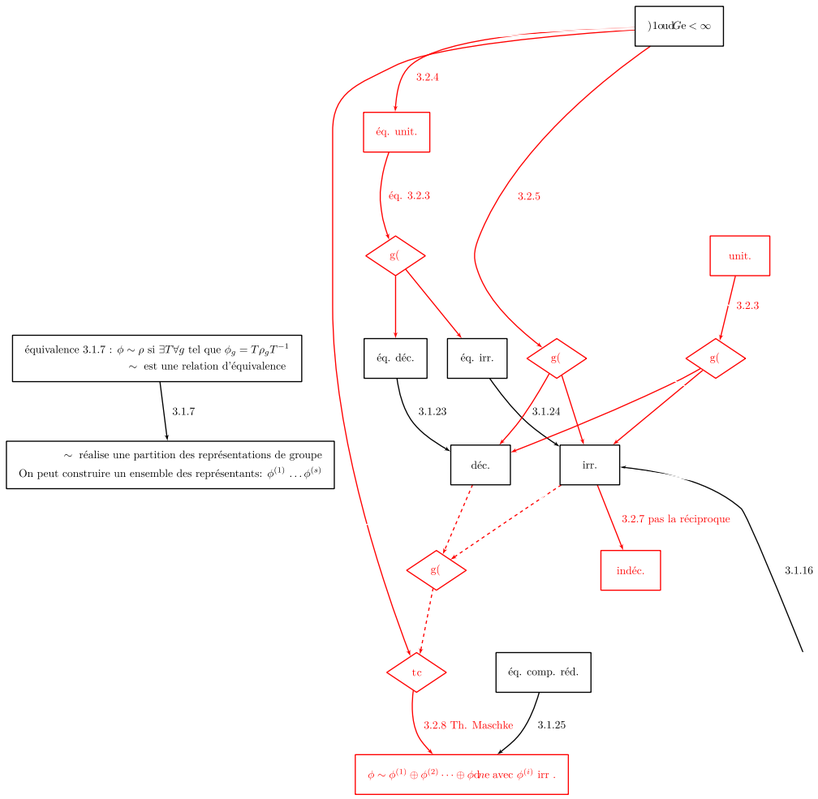
In [16]:
```
# Si le fichier est dans votre répertoire de travail
with open('week1.svg', 'r') as file:
    svg_content = file.read()

from IPython.display import HTML, display

def make_svg_responsive(svg_string):
    # Ajoute un viewBox s'il n'existe pas
    if 'viewBox' not in svg_string:
        import re
        width_match = re.search(r'width="(\d+)"', svg_string)
        height_match = re.search(r'height="(\d+)"', svg_string)

        if width_match and height_match:
            width = width_match.group(1)
            height = height_match.group(1)

            # Remplacer les attributs width/height par viewBox
            svg_string = svg_string.replace(
                f'width="{width}" height="{height}"',
                f'viewBox="0 0 {width} {height}"'
            )

    # Ajouter du style pour la responsivité
    svg_string = svg_string.replace(
        '<svg',
        '<svg style="width:100%;height:auto;max-width:100%;"'
    )

    return HTML(svg_string)

display(make_svg_responsive(svg_content))
```



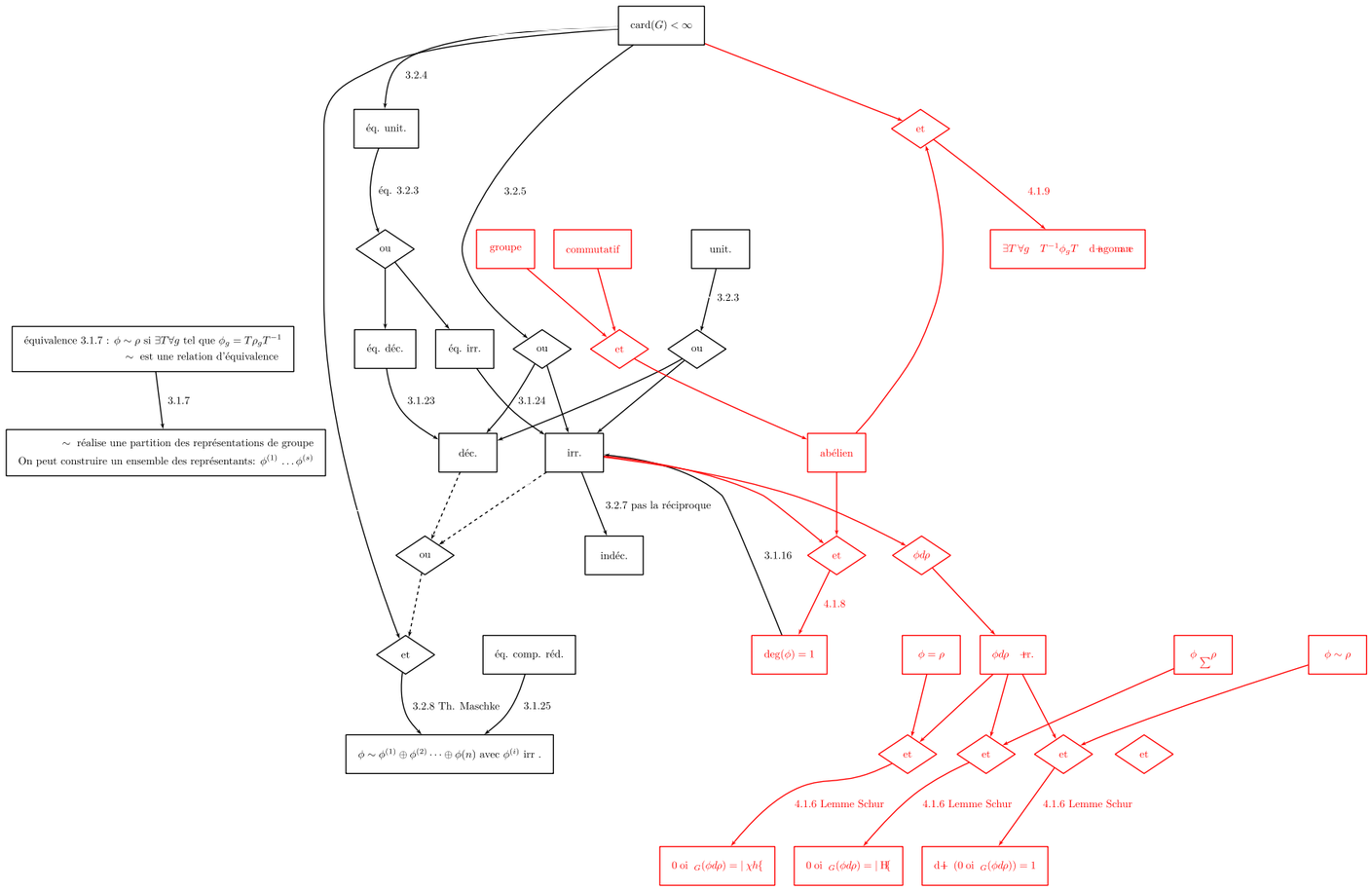
In [17]:
```
i=2

with open('week'+str(i)+'.svg', 'r') as file:
    svg_content = file.read()



display(make_svg_responsive(svg_content))
```



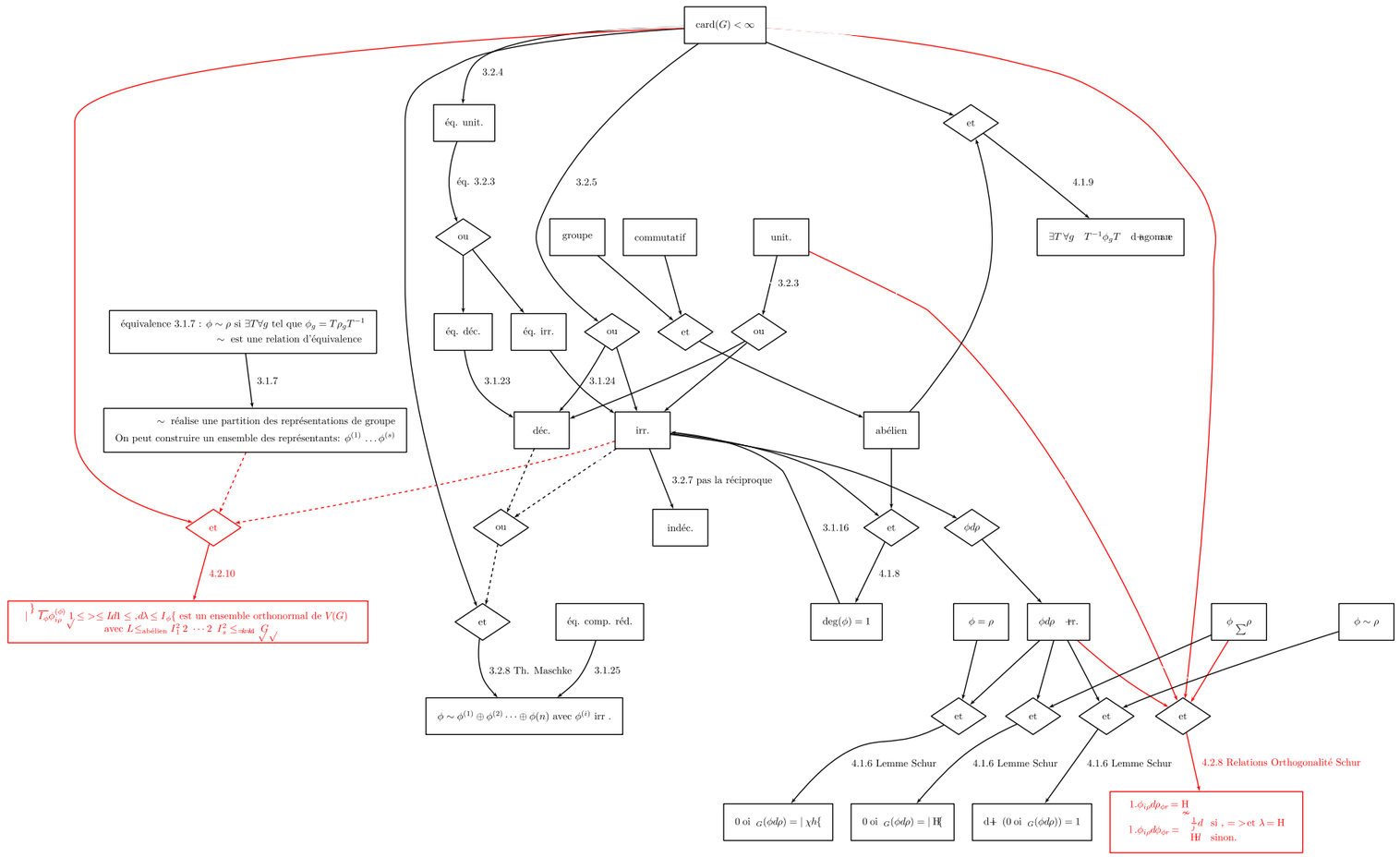
In [18]:
```
i=3

with open('week'+str(i)+'.svg', 'r') as file:
    svg_content = file.read()



display(make_svg_responsive(svg_content))
```



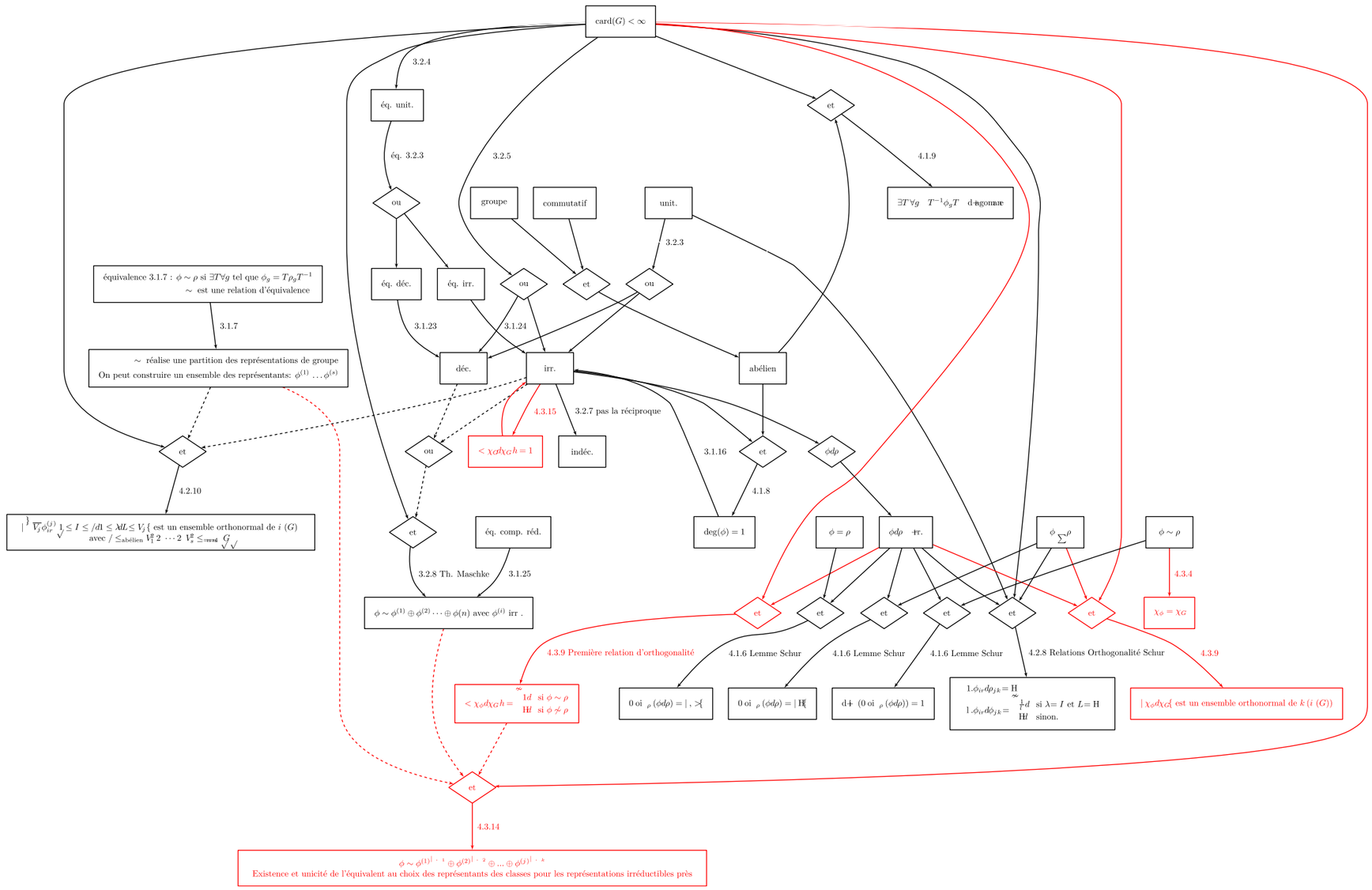
In [19]:
```
i=4

with open('week'+str(i)+'.svg', 'r') as file:
    svg_content = file.read()



display(make_svg_responsive(svg_content))
```



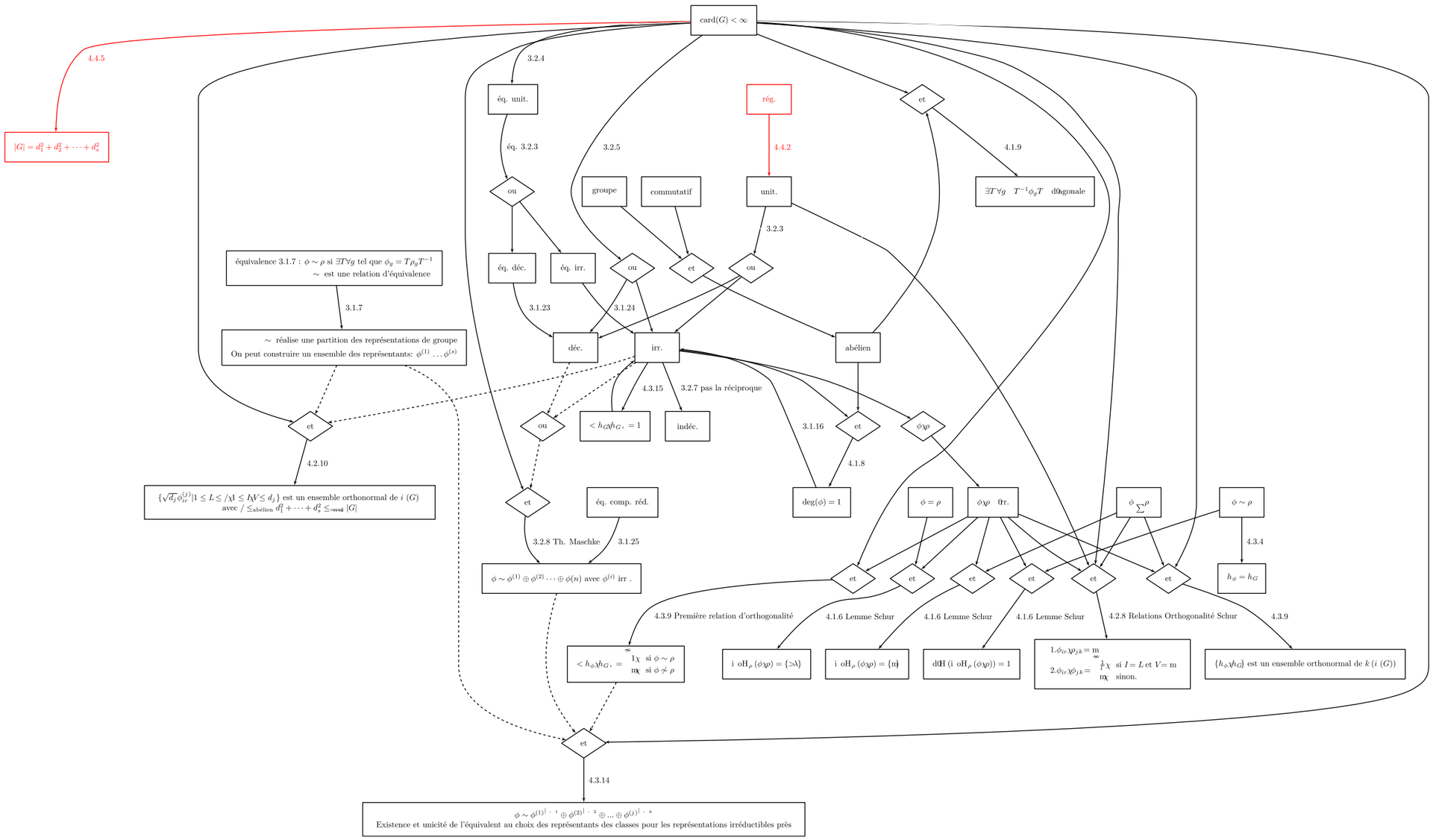
In [20]:
```
i=5

with open('week'+str(i)+'.svg', 'r') as file:
    svg_content = file.read()



display(make_svg_responsive(svg_content))
```



In [21]:
```
i=6

with open('week'+str(i)+'.svg', 'r') as file:
    svg_content = file.read()



display(make_svg_responsive(svg_content))
```



In [22]:
```
i=7

with open('week'+str(i)+'.svg', 'r') as file:
    svg_content = file.read()



display(make_svg_responsive(svg_content))
```



Théorème de représentation des groupes abéliens finis

Lemme de $\dim(L(G)) = |G|$

Nous allons utiliser des fonctions indicatrices de manière similaire à la démonstration de $\dim(Z(L(G)) = |\mathrm{Cl}(G)|$ sauf que nous allons utiliser les éléments du groupe comme références pour les fonctions au lieu des classes de conjugaison.

$\forall g \in G$, on définit la fonction $\delta_g : G \to \mathbb{C}$ par :

$$\delta_g(h) = \begin{cases} 1, & \text{si } h = g \\ 0, & \text{si } h \neq g \end{cases}$$

1. $\forall f \in L(G)$, on vérifie que:

$$f = \sum_{g \in G} f(g)\delta_g$$

En effet,

$$\forall h \quad \sum_{g \in G} f(g)\delta_g(h) = f(h)\delta_g(h) = f(h)$$

, car dans la somme, tous les termes $\delta_g(h)$ pour $g \neq h$ sont nuls. Donc $\{\delta_g\}$ génère $L(G)$.

2. Vérifions que $\{\delta_g\}$ est un ensemble orthogonal : $\forall h, h' \in G$, alors

$$\frac{1}{|G|} \sum_{g \in G} \delta_h(g)\overline{\delta_{h'}(g)} = \begin{cases} \frac{1}{|G|}, & \text{si } h = h' \\ 0, & \text{si } h \neq h' \end{cases}$$

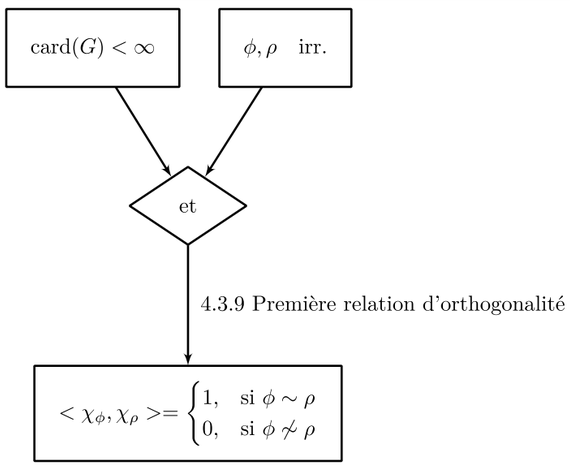
.

Sachant 1. et 2., $\{\delta_g | g \in G\}$ est une base de $L(G)$ et

$$\text{card}(\{\delta_g\}) = \text{card}(G) = \dim(L(G))$$

Lemme des représentations irréductibles de même traces

In [23]:
```
_=G_E(Grep,{"premrelortho"},1,0)
```



Par les premières relations d'orthogonalité pour les représentations irréductibles, on a l'équivalence:

$$\phi \sim \rho \overset{\text{irr}}{\Longleftrightarrow} \chi_\phi = \chi_\rho$$

Lemme des classes singletons en degré 1###

Si $\deg(\phi) = 1$ alors ϕ est irréductible, $\chi_\phi = \phi$ et on a les équivalences,

$$\phi \sim \rho \overset{\text{irr}}{\Longleftrightarrow} \chi_\phi = \chi_\rho \overset{\deg=1}{\Longleftrightarrow} \phi = \rho$$

Théorème 4.4.6: B.O.N. de $L(G)$

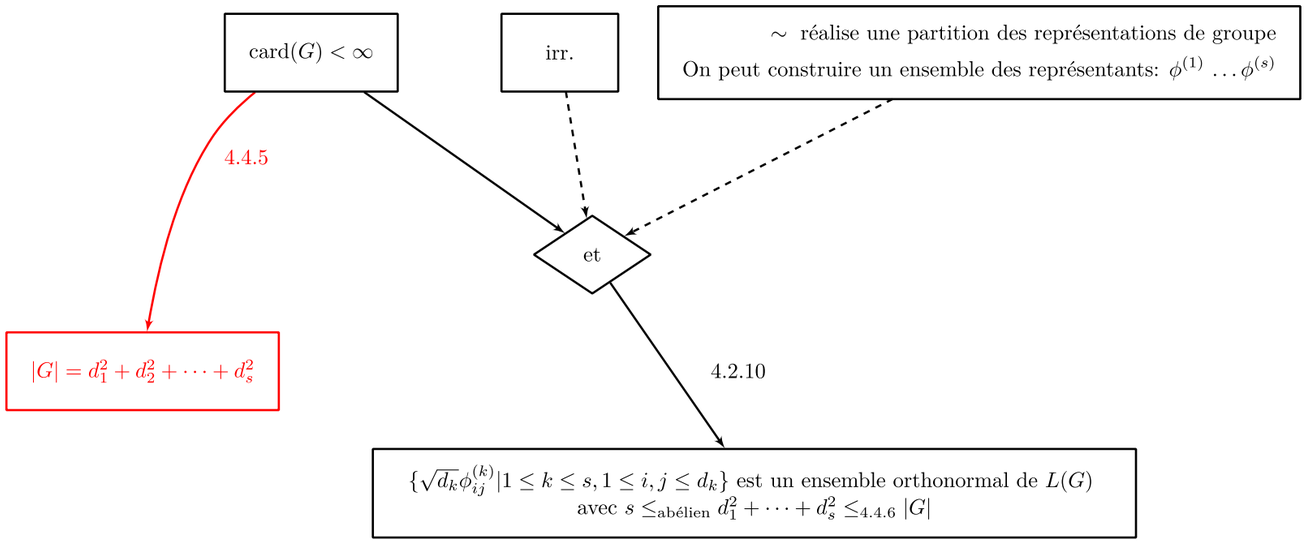
L'ensemble

$$B = \{\sqrt{d_k}\phi_{ij}^{(k)} | 1 \leq k \leq s, 1 \leq i, j \leq d_k\}$$

est une base orthonormée de $L(G)$.

Démonstration

In [24]:
```
_=G_E(Grep,{"orthosetLG"},2,2)
```



Il suffit de prouver que :

$$|B| = \dim(L(G))$$

On énumère le nombre d'éléments dans B (distincts car B est un ensemble orthonormal):

$$|B| = d_1^2 + d_2^2 + \cdots + d_s^2$$

Selon 4.4.5,

$$|G| = d_1^2 + d_2^2 + \cdots + d_s^2$$

Donc,

$$|B| = |G|$$

D'après le lemme sur la dimension de $L(G)$:

$$\dim(L(G)) = |G|$$

Finalememt,

$$|B| = \dim(L(G))$$

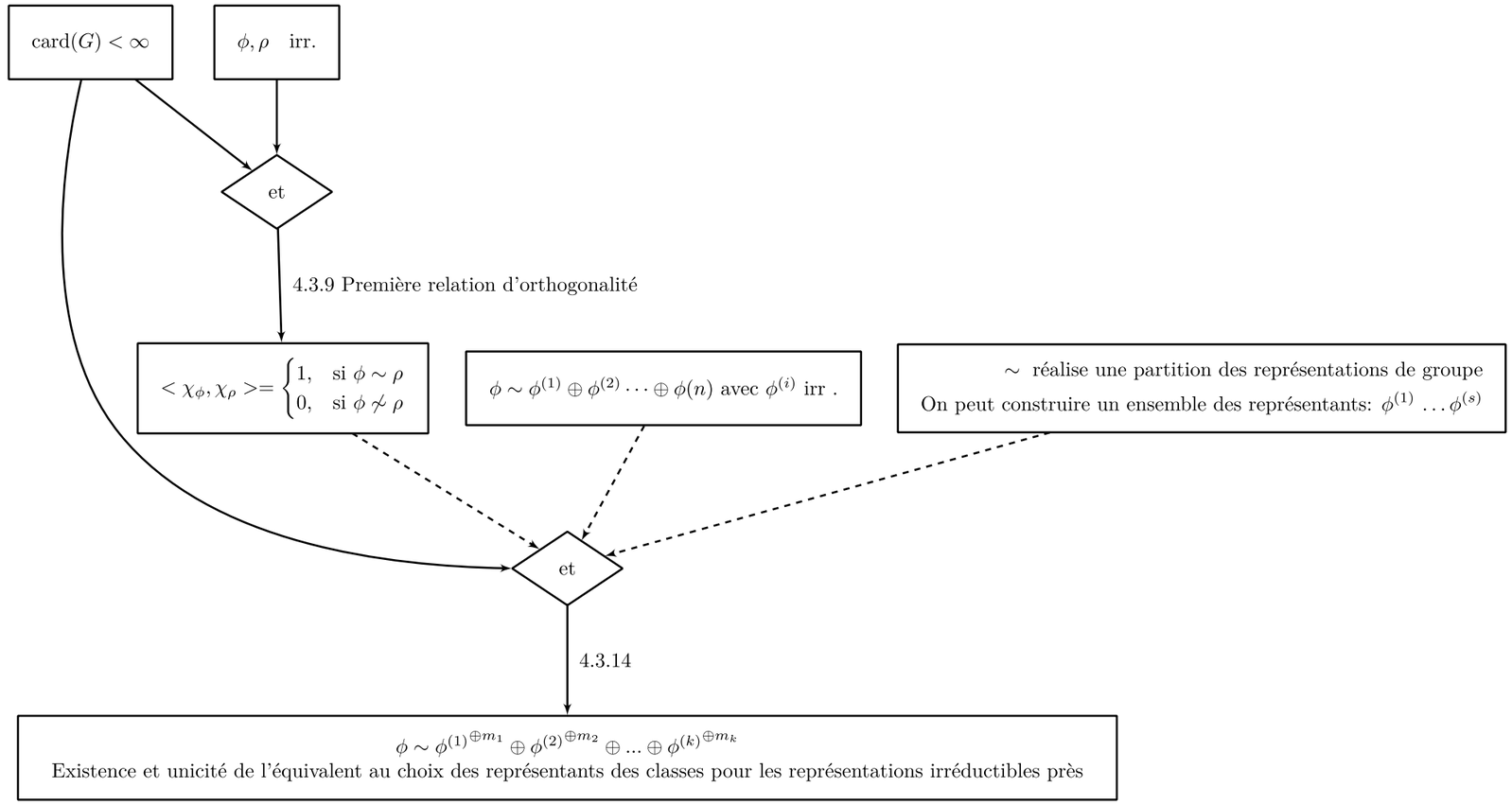
Théorème 4.4.7: B.O.N. de $Z(L(G))$

L'ensemble $\chi_1, \chi_2, \cdots, \chi_s$ est une base orthonormale de $Z(L(G))$.

Démonstration:

Rappel de notation : Fixons un ensemble complet $\{\varphi(1), \ldots, \varphi(s)\}$ de représentations unitaires irréductibles inéquivalentes de notre groupe fini G, et notons $d_i = \deg \varphi(i)$. Pour simplifier, posons $\chi_i = \chi_{\varphi(i)}$ pour $i = 1, \ldots, s$.

In [25]:
```
_=G_E(Grep,{"premrelortho"},2,2)
```



Selon la première relation d'orthogonalité, les $\{\chi_i\}$ forment un ensemble orthonormal. Il nous reste à prouver qu'ils génèrent $Z(L(G))$. $\forall f \in Z(L(G)), \quad \forall x \in G$, en décomposant sur la base orthonormée trouvée précédemment:

$$
\begin{aligned}
f(x) &= \frac{1}{|G|}\sum_{g\in G} f(g^{-1}xg) && \cdot_{\text{class}} : L(G) \to Z(L(G)) \text{ et } f \in Z(L(G)) \text{ donc } f_{\text{classe}} = f && (1)\\
&= \frac{1}{|G|}\sum_{g\in G}\sum_{i,j,k} c_{ij}^{(k)}\phi_{ij}^{(k)}(g^{-1}xg) && \text{décomposition sur B.O.N.} && (2)\\
&= \sum_{i,j,k} c_{ij}^{(k)}\left[\frac{1}{|G|}\sum_{g\in G}\phi_{ij}^{(k)}(g^{-1}xg)\right] && \text{on permute les deux signes somme} && (3)\\
&= \sum_{i,j,k} c_{ij}^{(k)}\left[\frac{1}{|G|}\sum_{g\in G}\phi_{g^{-1}}^{(k)}\phi_{x}^{(k)}\phi_{g}^{(k)}\right]_{ij} && \phi^k \text{ est un morphisme} && (4)\\
&= \sum_{i,j,k} c_{ij}^{(k)}\left[\left(\phi_x^{(k)}\right)^{\#}\right]_{ij} && \text{Formule 4.2.2 (a) avec } T = \phi_x^{(k)} && (5)\\
&= \sum_{i,j,k} c_{ij}^{(k)}\frac{Tr(\phi^{(k)}x)}{\deg\phi^{(k)}}\delta_{i,j} && \text{Formule 4.2.3 (b) avec } \phi = \rho && (6)\\
&= \sum_{i,k} c_{ii}^{(k)}\frac{1}{d_k}\chi^{(k)}(x) && \text{formule de } \chi_k(x) && (7)
\end{aligned}
$$

Les $\chi^{(k)}$ engendrent bien $Z(L(G))$. Les $\chi^{(k)}$ forment une famille orthonormale donc libre. On a bien une base. On en déduit ainsi que $\dim(Z(L(G)) = s$.

Corollaire 4.4.8 $|\mathrm{Hom}_{irr}(G,V)/\sim| = |G/\sim|$

Par construction des χ_k:

$$s = |\mathrm{Hom}_{irr}(G,V)/\sim|$$

D'après le théorème précédent:

$$s = \dim(Z(L(G))$$

D'après 4.3.8 (les δ_C forment aussi une B.O.N.):

$$\dim(Z(L(G)) = |G/\sim|$$

Corollaire 4.4.9 G.A.F. $\iff |G| = |\mathrm{Hom}_{irr}(G,V)/\sim|$

Groupe abélien fini $\iff |G| = |G/\sim|$
Il suffit d'appliquer 4.4.8 .

Exemple 4.4.10 $\mathrm{Hom}_{irr}(\mathbb{Z}/n\mathbb{Z},\mathbb{C}^*) = \{e^{\frac{2\pi ik\cdot}{n}}\}_{k\in[\![0,n-1]\!]}$

Définissons $\forall [m]$:

$$\chi_k([m]) = e^{\frac{2\pi ikm}{n}}$$

Comme $\chi_k([m+p]) = \chi_k([m])\chi_k([p])$, on a bien un homomorphisme et $\chi_k([m]) \in \mathbb{C}^*$.
$\deg(\chi_k) = 1$ donc les χ_k sont irréductibles. \

Si $\chi_k([m]) = \chi_l([m])$ alors:

$$e^{\frac{2\pi ikm}{n}} = e^{\frac{2\pi ilm}{n}}$$

$$\frac{2\pi km}{n} = \frac{2\pi lm}{n} + 2j\pi \text{ avec } j \in \mathbb{Z}$$

$$(k-l)m = jn$$

Si $j = 0$, en prenant $m = 1$, $k - l = 0$.
Si $j \neq 0$, on a $n|(k-l)m$ et en prenant $m = n-1$, on a $pgcd(m,n) = 1$, on peut appliquer le lemme de Gauss et donc $n|k-l$. Comme $|k-l| \leq n-1$, on a bien $k-l=0$.
Ils sont tous distincts, au nombre de n et $\mathbb{Z}/n\mathbb{Z}$ est abélien fini donc ce sont **les** représentations irréductibles de $\mathbb{Z}/n\mathbb{Z}$.

In [26]:
```
n = 4
result = create_array_ZnZ(n)
latex_table=array_to_latex_with_bold(result)

display(Markdown(f"$${latex_table}$$"))
```

	$[\mathbf{0}]$	$[\mathbf{1}]$	$[\mathbf{2}]$	$[\mathbf{3}]$
$\chi_{\mathbf{0}}$	$(1+0j)$	$(1+0j)$	$(1+0j)$	$(1+0j)$
$\chi_{\mathbf{1}}$	$(1+0j)$	$1j$	$(-1+0j)$	$(-0-1j)$
$\chi_{\mathbf{2}}$	$(1+0j)$	$(-1+0j)$	$(1-0j)$	$(-1+0j)$
$\chi_{\mathbf{3}}$	$(1+0j)$	$(-0-1j)$	$(-1+0j)$	$1j$

In [27]:
```
# Example usage:
n = 2
result = create_array_ZnZ(n)
latex_table=array_to_latex_with_bold(result)

display(Markdown(f"$${latex_table}$$"))
```

	$[\mathbf{0}]$	$[\mathbf{1}]$
$\chi_{\mathbf{0}}$	$(1+0j)$	$(1+0j)$
$\chi_{\mathbf{1}}$	$(1+0j)$	$(-1+0j)$

In [28]:
```
# Example usage:
n = 8
result = create_array_ZnZ(n)
latex_table=array_to_latex_with_bold(result)

display(Markdown(f"$${latex_table}$$"))
```

	$[\mathbf{0}]$	$[\mathbf{1}]$	$[\mathbf{2}]$	$[\mathbf{3}]$	$[\mathbf{4}]$	$[\mathbf{5}]$	$[\mathbf{6}]$	$[\mathbf{7}]$
$\chi_{\mathbf{0}}$	$(1+0j)$	$(1+0j)$	$(1+0j)$	$(1+0j)$	$(1+0j)$	$(1+0j)$	$(1+0j)$	$(1+0j)$
$\chi_{\mathbf{1}}$	$(1+0j)$	$(1+1j)$	$1j$	$(-1+1j)$	$(-1+0j)$	$(-1-1j)$	$(-0-1j)$	$(1-1j)$
$\chi_{\mathbf{2}}$	$(1+0j)$	$1j$	$(-1+0j)$	$(-0-1j)$	$(1-0j)$	$1j$	$(-1+0j)$	$(-0-1j)$
$\chi_{\mathbf{3}}$	$(1+0j)$	$(-1+1j)$	$(-0-1j)$	$(1+1j)$	$(-1+0j)$	$(1-1j)$	$1j$	$(-1-1j)$
$\chi_{\mathbf{4}}$	$(1+0j)$	$(-1+0j)$	$(1-0j)$	$(-1+0j)$	$(1-0j)$	$(-1+0j)$	$(1-0j)$	$(-1+0j)$
$\chi_{\mathbf{5}}$	$(1+0j)$	$(-1-1j)$	$1j$	$(1-1j)$	$(-1+0j)$	$(1+1j)$	$(-0-1j)$	$(-1+1j)$
$\chi_{\mathbf{6}}$	$(1+0j)$	$(-0-1j)$	$(-1+0j)$	$1j$	$(1-0j)$	$(-0-1j)$	$(-1+0j)$	$(-0+1j)$
$\chi_{\mathbf{7}}$	$(1+0j)$	$(1-1j)$	$(-0-1j)$	$(-1-1j)$	$(-1+0j)$	$(-1+1j)$	$(-0+1j)$	$(1+1j)$

Théorème 4.4.13 Seconde relation d'orthogonalité

Soient C et C' des classes de conjugaison de G, et soient $g \in C$ et $h \in C'$. Alors :

$$\sum_{i=1}^{s} \chi_i(g)\overline{\chi_i(h)} = \begin{cases} \frac{|G|}{|C|} & \text{si } C = C', \\ 0 & \text{si } C \neq C'. \end{cases}$$

En conséquence, les colonnes de la table des caractères sont orthogonales, ce qui rend la table des caractères inversible.

Démonstration:

Nous calculons :

$$\delta_{C'}(g) = \sum_{i=1}^{s} \langle \delta_{C'}, \chi_i \rangle \chi_i(g) \qquad \text{décomposition sur la B.O.N. des } \delta_C \tag{8}$$

$$= \sum_{i=1}^{s} \frac{1}{|G|} \sum_{x \in G} \delta_{C'}(x) \overline{\chi_i(x)} \chi_i(g) \qquad \text{écriture du produit scalaire} \tag{9}$$

$$= \sum_{i=1}^{s} \frac{1}{|G|} \sum_{x \in C'} \overline{\chi_i(x)} \chi_i(g) \qquad \delta_{C'} \text{passe dans les indices de } \sum \tag{10}$$

$$= \frac{|C'|}{|G|} \sum_{i=1}^{s} \chi_i(g) \overline{\chi_i(h)} \qquad \chi_i \in Z(L(G)) \tag{11}$$

Puisque le membre de gauche vaut 1 lorsque $g \in C'$ et 0 sinon, nous concluons que

$$\sum_{i=1}^{s} \overline{\chi_i(h)} \chi_i(g) = \begin{cases} \frac{|G|}{|C'|} & \text{si } C = C' \\ 0 & \text{si } C \neq C' \end{cases}$$

comme requis.

Remarque 4.4.13 matrice de passage $\{\chi_i\} \to \{\delta_C\}$

La matrice de passage a, par définition, comme colonne les vecteurs de la nouvelle base dans les coordonnées de l'ancienne base. \

En appliquant la dernière formule trouvée avec $h \in C'$:

$$\delta_{C'}(\cdot) = \frac{|C'|}{|G|} \sum_{i=1}^{s} \chi_i(\cdot) \overline{\chi_i(h)}$$

Donc les coefficients de $P = p_{i,j}$ sont

$$p_{i,j} = \frac{|C_j|}{|G|} \overline{\chi_i(C_j)}$$

Les coefficients du tableau de caractère sont :

$$\chi_i(C_j)$$

Rappel : théorème de décomposition des groupes abéliens finis

Tout groupe abélien fini G est isomorphe à un produit direct de groupes cycliques de la forme :

$$\mathbb{Z}_{p_1^{\alpha_1}} \times \mathbb{Z}_{p_2^{\alpha_2}} \times \cdots \times \mathbb{Z}_{p_n^{\alpha_n}}$$

où les p_i sont des nombres premiers (pas nécessairement distincts).

Propriété 4.5.1 produit cartésien et produit de caractère

(4.5.1)
Soit G_1,G_2 des groupes abéliens.
$\chi_1, \ldots, \chi_m$ et $\phi_1, \ldots, \phi_n$ sont les représentations irréductibles de G_1 et G_2, respectivement.
En particulier, $m = |G_1|$ et $n = |G_2|$.
Alors les fonctions $\alpha_{ij} : G_1 \times G_2 \to \mathbb{C}^*$ avec $1 \leq i \leq m$ et $1 \leq j \leq n$ donnée par:

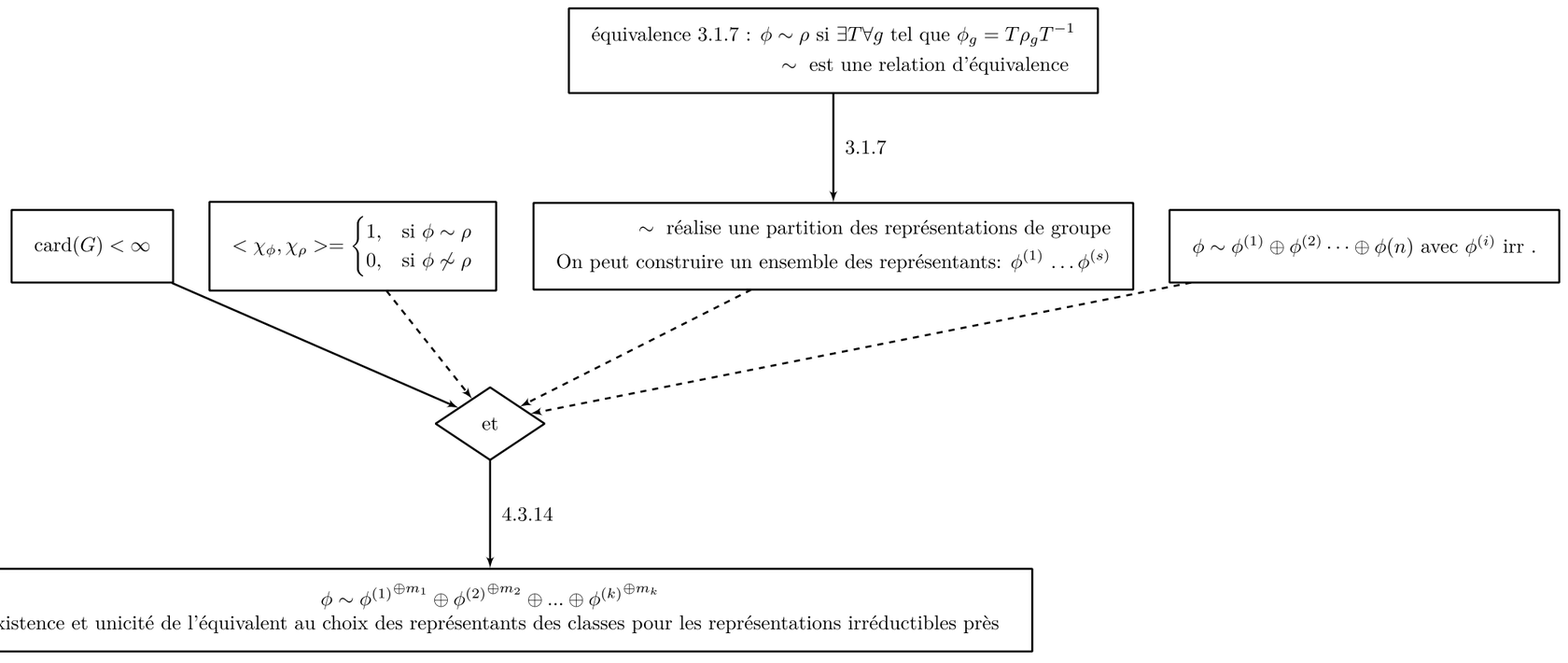
$$\alpha_{ij}(g_1, g_2) = \chi_i(g_1)\phi_j(g_2) \tag{1}$$

forment un ensemble complet de représentations irréductibles de $G_1 \times G_2$.

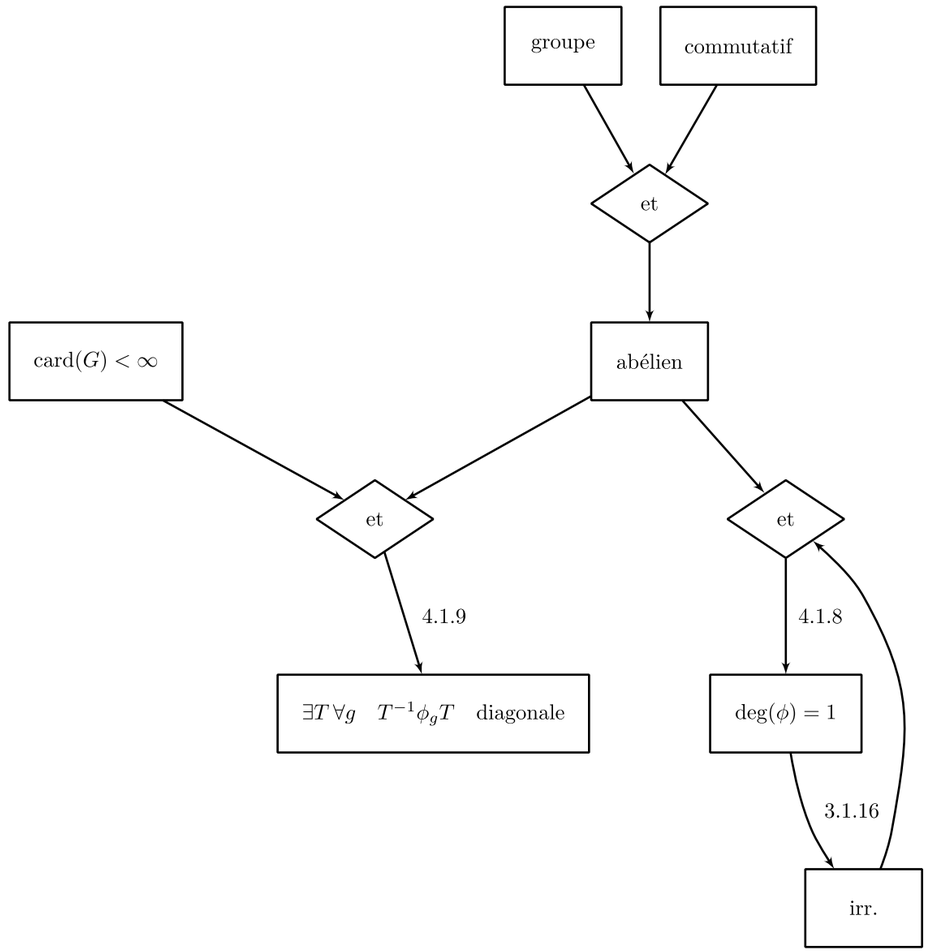
Démonstration

1. $\chi_1, \ldots, \chi_m$ *et* $\phi_1, \ldots, \phi_n$ *sont les représentations irréductibles de* G_1 *et* G_2*, respectivement. En particulier,* $m = |G_1|$ *et* $n = |G_2|$. \

In [29]:
```
_=G_E(Grep,{'unique_dec'},1,0,({'classe_eq'},1,0),({'comp_red'},0,2))
```



In [30]:
```
_=G_E(Grep,{"abelien"},2,2)
```



Par le lemme des classes singletons, ce sont les seuls. En effet, la représentation triviale (3.1.3) appartient à $\mathrm{Hom}_{irr}(G, \mathbb{C}^*)$. Donc cet ensemble est non vide. Prenons $\sim$ pour réaliser une partition de $\mathrm{Hom}_{irr}(G, \mathbb{C}^*)$. Soit ρ une représentation irréductible. Donc ρ appartient forcément à une des classes d'équivalence (disons la i). $\deg(\rho) = 1$ car ρ est irréductible et G est abélien et fini. En utilisant le lemme:

$$\rho \sim \chi_i \iff \chi_\rho = \chi_i \iff \rho = \phi^{(i)}$$

On a bien existence et unicité de l'ensemble des représentants.

3. $\alpha_{ij} \in \mathrm{Hom}_{irr}(G_1 \times G_2, \mathbb{C}^*)$

Le produit cartésien est muni d'une loi de composition interne $\bullet_{G_1 \times G_2}$:
$(g_1, g_1') \bullet_{G_1 \times G_2} (g_2, g_2') = (g_1 \bullet_{G_1} g_2 \ , \ g_1' \bullet_{G_2} g_2') \setminus$

$\forall (g_1, g_2) \in G_1 \times G_2$ et $\forall (g_1', g_2') \in G_1 \times G_2$:

$$\begin{aligned}
\alpha_{ij}(g_1, g_2)\alpha_{ij}(g_1', g_2') &= \chi_i(g_1)\phi_j(g_2)\chi_i(g_1')\phi_j(g_2') \\
&= \chi_i(g_1)\chi_i(g_1')\phi_j(g_2)\phi_j(g_2') \\
&= \chi_i(g_1 g_1')\phi_j(g_2 g_2') \\
&= \alpha_{ij}(g_1 g_1', g_2 g_2') \\
&= \alpha_{ij}((g_1, g_2)(g_1', g_2'))
\end{aligned}$$

$\deg(\alpha_{ij}) = 1$ donc d'après 3.1.16, α_{ij} est irréductible.

4. Les α_{ij} sont distincts:

Si $\alpha_{ij} = \alpha_{kl}$, sachant que $\forall m\ \chi_m(1) = \phi_m(1) = 1$ selon 4.3.3 et en prenant $\forall g \in G_1$: \

$\chi_i(g) = \chi_i(g)\phi_j(1) = \alpha_{ij}(g,1) = \alpha_{kl}(g,1) = \chi_k(g)\phi_l(1) = \chi_k(g)$ \

Donc $\chi_i = \chi_k$, et comme les χ sont distincts, $i = k$. \

Même raisonnement avec $\phi_j = \phi_l$ en prenant $(1, g)$. \

5. Égalité des ensembles

$|G_1 \times G_2| = mn$ donc $G_1 \times G_2$ a mn classes d'équivalence de représentations irréductibles selon 4.4.9 . \

$\text{card}(\alpha_{ij}) = mn$ par construction et car les α_{ij} sont distincts. \

$\alpha_{ij} \subseteq \text{Hom}_{irr}(G_1 \times G_2, \mathbb{C}^*)$ \

Par le lemme des classes singletons, chaque α_{ij} est l'unique représentant de sa classe. \

Donc les α_{ij} forment **l'ensemble** complet des représentations irréductibles.

Exemple 4.5.2 :

In [31]:
```
perm = [1, 0, 2, 3]  # Permutation des indices
matrice = matrice_permutation(perm)
print(matrice)
```

```
[[0 1 0 0]
 [1 0 0 0]
 [0 0 1 0]
 [0 0 0 1]]
```

Groupe de Klein

In [32]:
```
import numpy as np

def groupe_klein():
    """
    Définit explicitement les éléments du groupe de Klein.

    Returns:
    list: Liste des 4 éléments du groupe
    """
    # Éléments du groupe de Klein









    e = np.array([[1, 0], [0, 1]])             # Identité
    a = np.array([[1, 0], [0, -1]])           # Réflexion horizontale
    b = np.array([[-1, 0], [0, 1]])          # Réflexion verticale
```

```
    c = np.array([[-1, 0], [0, -1]])          # Symétrie centrale

    return [e, a, b, c]
(e,a,b,c)=groupe_klein()
name=['e','a','b','c']
name=np.array(name)



def matrices_equal(m1, m2, tol=1e-8):
    """
    Compare deux matrices en tenant compte des erreurs de précision numérique.

    Args:
    m1, m2 (np.ndarray): Matrices à comparer.
    tol (float): Tolérance pour les différences.

    Returns:
    bool: True si les matrices sont égales, False sinon.
    """
    return np.allclose(m1, m2, atol=tol)




def table_multiplication(elements = groupe_klein(),name=name):
    """
    Génère la table de multiplication du groupe de Klein.
    """

    n = len(elements)
    table = np.zeros((n, n),dtype=str)


    for i in range(n):
        row = f" {i} |"
        for j in range(n):
            produit = np.dot(elements[i], elements[j])
            test=[matrices_equal(produit,elements[k]) for k in range(len(elements))]
            table[i, j] = name[test][0]


    table=np.array(table)
    table=np.vstack((name,table))
    table=np.hstack((np.transpose(np.concatenate((['\\ast'],name))).reshape(5, 1),table))
    return table

# Générer et afficher la table de multiplication
table = table_multiplication()
latex_table = array_to_latex_with_bold(table)
display(Markdown("Représentation du groupe $D_4$ dans $GL(\mathbb{R}^2)$:"))
display(Markdown(f"e : ${array_to_latex(e)}$ $\quad$  a : ${array_to_latex(a)}$ "))
display(Markdown(f"b:  ${array_to_latex(b)}$   $\quad$   c: ${array_to_latex(c)}$"))
display(Markdown("Table de Caley de $D_4$:"))
display(Markdown(f"$${latex_table}$$"))
```

Représentation du groupe D_4 dans $GL(\mathbb{R}^2)$:

e : $\begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & 1 \\ \hline \end{array}$ a : $\begin{array}{|c|c|}\hline 1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}$

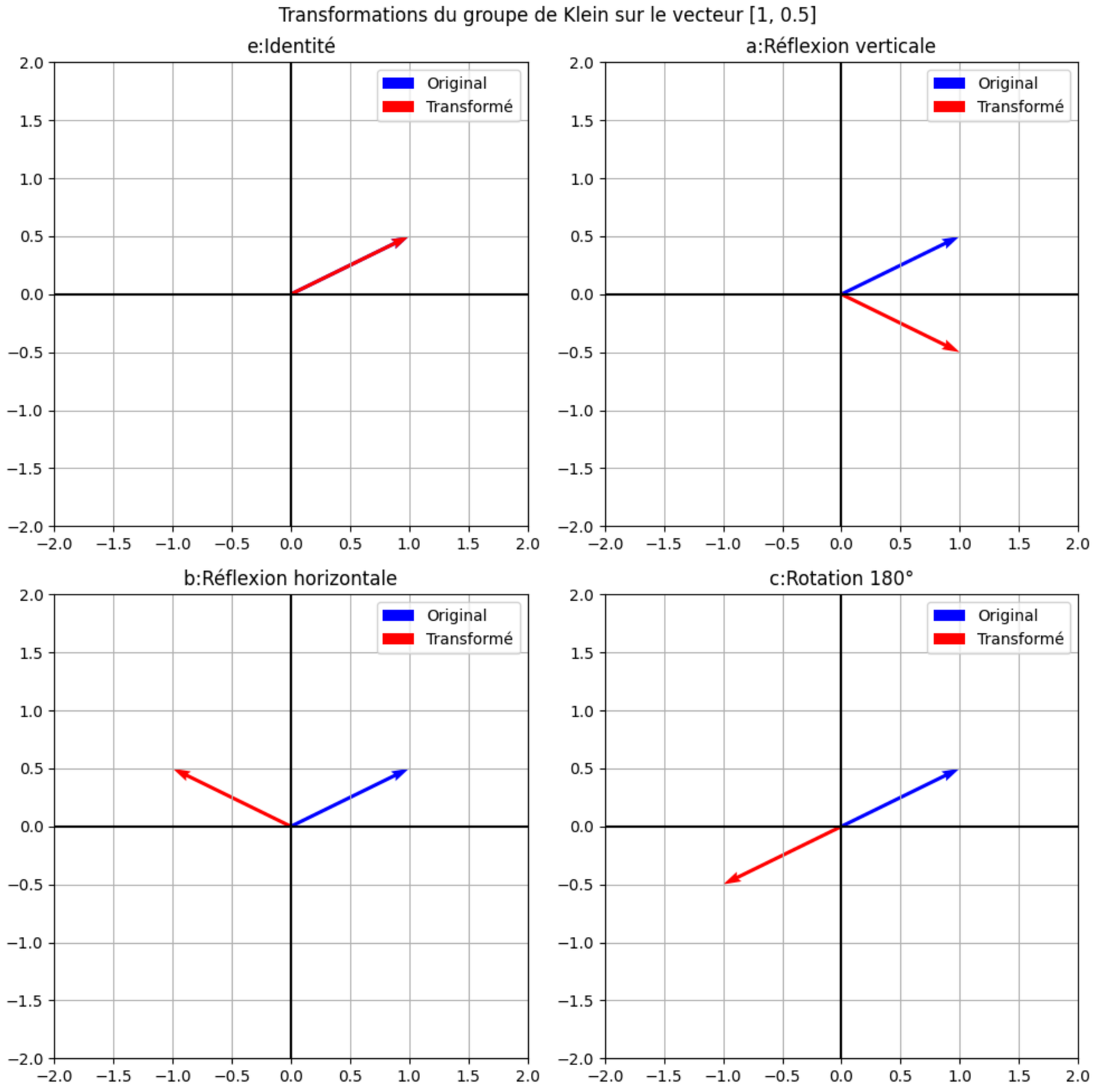
b: $\begin{array}{|c|c|}\hline -1 & 0 \\ \hline 0 & 1 \\ \hline \end{array}$ c: $\begin{array}{|c|c|}\hline -1 & 0 \\ \hline 0 & -1 \\ \hline \end{array}$

Table de Caley de D_4:

$*$	**e**	**a**	**b**	**c**
e	e	a	b	c
a	a	e	c	b
b	b	c	e	a
c	c	b	a	e

In [33]:
```python
import numpy as np
import matplotlib.pyplot as plt

# Définition des matrices de transformation
e = np.array([[1, 0], [0, 1]])             # Identité
a = np.array([[1, 0], [0, -1]])            # Réflexion verticale
b = np.array([[-1, 0], [0, 1]])            # Réflexion horizontale
c = np.array([[-1, 0], [0, -1]])           # Rotation 180°

# Vecteur initial
v = np.array([1, 0.5])

# Créer une figure en mosaïque
fig, axs = plt.subplots(2, 2, figsize=(10, 10))
fig.suptitle('Transformations du groupe de Klein sur le vecteur [1, 0.5]')

# Liste des transformations et leurs noms
transformations = [
    (e, 'e:Identité'),
    (a, 'a:Réflexion verticale'),
    (b, 'b:Réflexion horizontale'),
    (c, 'c:Rotation 180°')
]

# Appliquer chaque transformation
for i, (mat, name) in enumerate(transformations):
    # Calculer la transformation
    transformed = mat @ v

    # Déterminer la position dans la grille
    row = i // 2
    col = i % 2

    # Tracer le vecteur original et transformé
    axs[row, col].quiver(0, 0, v[0], v[1], angles='xy', scale_units='xy', scale=1, color='blue', label='Original')
    axs[row, col].quiver(0, 0, transformed[0], transformed[1], angles='xy', scale_units='xy', scale=1, color='red', label='Transformé')

    axs[row, col].set_xlim(-2, 2)
    axs[row, col].set_ylim(-2, 2)
    axs[row, col].set_title(name)
    axs[row, col].grid(True)
    axs[row, col].axhline(y=0, color='k')
```

```
    axs[row, col].axvline(x=0, color='k')
    axs[row, col].legend()

plt.tight_layout()
plt.show()
```



isométries laissant globalement invariant un rectangle (éventuellement réduit à un segment) : D_4

$$\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$$

In [34]:
```
data = {
    "": ["\chi_1", "\chi_2"],
    "[0]": [1, 1 ],
    "[1]": [1, -1]
}
df = pd.DataFrame(data)
datachi= {    "": ["\chi_1", "\chi_2"],
    "[0]": ["\chi_1([0])", "\chi_2([0])"],
  "[1]": ["\chi_1([1])", "\chi_2([1])"]}
dfchi = pd.DataFrame(datachi)
latex_table = dataframe_to_latex_array(df)
latex_table_chi = dataframe_to_latex_array(dfchi)

# Display LaTeX code in Markdown
display(Markdown("Tableau des caractères"))
display(Markdown(f"$${latex_table_chi}$$"))

# Display LaTeX code in Markdown
display(Markdown("Tableau des caractères de $\mathbb{Z}/2\mathbb{Z}$"))
display(Markdown(f"$${latex_table}$$"))
```

Tableau des caractères

	$[0]$	$[1]$
χ_1	$\chi_1([0])$	$\chi_1([1])$
χ_2	$\chi_2([0])$	$\chi_2([1])$

Tableau des caractères de $\mathbb{Z}/2\mathbb{Z}$

	$[0]$	$[1]$
χ_1	1	1
χ_2	1	-1

In [35]:
```
data = {
    "": ["\\alpha_{11}", "\\alpha_{12}", "\\alpha_{21}", "\\alpha_{22}"],
    "([0],[0])": [1, 1,1,1 ],
    "([0],[1])":  [1, -1,1,-1],
    "([1],[0])": [1,1,-1 ,-1],
    "([1],[1])": [1,-1,-1,1]
}
df = pd.DataFrame(data)

latex_table = dataframe_to_latex_array(df)
npchi2 = dfchi.to_numpy()
#display(npchi2)

def string_kronecker_product(arr1, arr2):
    """
    Perform a Kronecker product-like operation on string arrays
    by concatenating elements
    """
```

```
    result = np.empty((arr1.shape[0] * arr2.shape[0],
                       arr1.shape[1] * arr2.shape[1]),
                      dtype=object)

    for i in range(arr1.shape[0]):
        for j in range(arr1.shape[1]):
            for k in range(arr2.shape[0]):
                for l in range(arr2.shape[1]):
                    result[i*arr2.shape[0] + k, j*arr2.shape[1] + l] = \
                        f"{arr1[i,j]}{arr2[k,l]}"

    return result

# Example usage:
npchi2 = np.array([[ '\\chi_1([0])', '\\chi_1([1])'],
                   [ '\\chi_2([0])', '\\chi_2([1])']], dtype=object)

npchi2_red= np.array([[ '\\textcolor{red}{\\chi_1([0])}', '\\textcolor{red}{\\chi_1([1])}'],
                   [ '\\textcolor{red}{\\chi_2([0])}', '\\textcolor{red}{\\chi_2([1])}']], dtype=object)


npchi4 = string_kronecker_product(npchi2_red, npchi2)

npchi4=np.vstack((['([0],[0])','([0],[1])','([1],[0])','([1],[1])'],npchi4))
npchi4=np.hstack((np.transpose(['','\\chi_1\\chi_1','\\chi_1\\chi_2','\\chi_2\\chi_1','\\chi_2\\chi_2']).reshape(5,1),npchi4))
ltxchi4=array_to_latex(npchi4)
display(Markdown("Tableau des caractères de $\mathbb{Z}/2\mathbb{Z} \\times \mathbb{Z}/2\mathbb{Z}$"))
display(Markdown(f"$${ltxchi4}$$"))# Display LaTeX code in Markdown
display(Markdown("Tableau des caractères de $\mathbb{Z}/2\mathbb{Z} \\times \mathbb{Z}/2\mathbb{Z}$"))
display(Markdown(f"$${latex_table}$$"))
```

Tableau des caractères de $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$

	$([0],[0])$	$([0],[1])$	$([1],[0])$	$([1],[1])$
$\chi_1\chi_1$	$\chi_1([0])\chi_1([0])$	$\chi_1([0])\chi_1([1])$	$\chi_1([1])\chi_1([0])$	$\chi_1([1])\chi_1([1])$
$\chi_1\chi_2$	$\chi_1([0])\chi_2([0])$	$\chi_1([0])\chi_2([1])$	$\chi_1([1])\chi_2([0])$	$\chi_1([1])\chi_2([1])$
$\chi_2\chi_1$	$\chi_2([0])\chi_1([0])$	$\chi_2([0])\chi_1([1])$	$\chi_2([1])\chi_1([0])$	$\chi_2([1])\chi_1([1])$
$\chi_2\chi_2$	$\chi_2([0])\chi_2([0])$	$\chi_2([0])\chi_2([1])$	$\chi_2([1])\chi_2([0])$	$\chi_2([1])\chi_2([1])$

Tableau des caractères de $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$

	$([0],[0])$	$([0],[1])$	$([1],[0])$	$([1],[1])$
α_{11}	1	1	1	1
α_{12}	1	-1	1	-1
α_{21}	1	1	-1	-1
α_{22}	1	-1	-1	1

Applications

Exercice 4.9

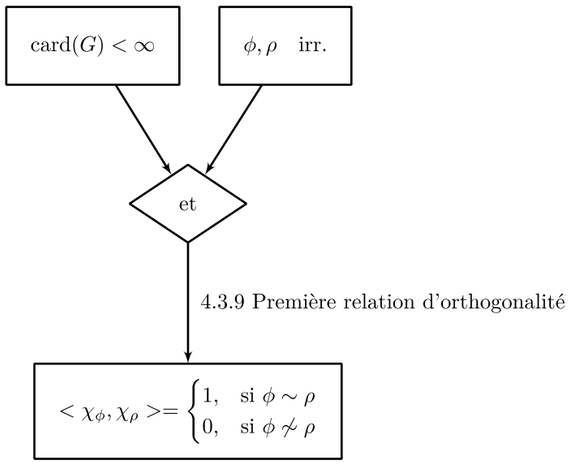
Énoncé :

Soit χ un caractère irréductible non trivial d'un groupe fini G. Montrer que :

$$\sum_{g\in G}\chi_\rho(g)=0$$

Résolution :

In [36]:
```
_=G_E(Grep,{"premrelortho"},2,0)
```



$$\langle\chi_\rho,\chi_{\text{trivial}}\rangle=0$$

$$\frac{1}{|G|}\sum_{g\in G}\chi_\rho(g)\overline{\chi_{\text{trivial}}(g)}=0$$

Et comme $\forall g\ \ \chi_{\text{trivial}}(g)=1$,on a bien :

$$\sum_{g\in G}\chi_\rho(g)=0$$

Exercice 4.7

Énoncé :

Soient $\phi : G\to \mathrm{GL}_n(\mathbb{C})$ et $\rho : G\to \mathrm{GL}_m(\mathbb{C})$ deux représentations. Soit $V=M_{mn}(\mathbb{C})$.

Définissons $\tau : G\to \mathrm{GL}(V)$ par $\tau_g(A)=\rho_g A\phi_g^T$.

1.Montrer que τ est une représentation de G.

Résolution :

Nous devons vérifier que c'est un homomorphisme de groupe. $\forall A\in M_{mn}(\mathbb{C})$ et $\forall g,h\in G$:

$$\tau_{gh}(A)=\rho_{gh}A\phi_{gh}^T=\rho_g(\rho_hA\phi_h^T)\phi_g^T=\tau_g(\tau_h(A))$$

On en déduit aussi, comme pour tout homomorphisme :

$$\tau_e(A) = \rho_e A \phi_e^T = A$$

et

$$\tau_{g^{-1}}(A) = (\tau_g(A))^{-1}$$

Énoncé :

Soit τ la représentation définie précédemment.
2.Montrer que :

$$\tau_g(E_{kl}) = \sum_{i,j} \rho_{ik}(g)\phi_{jl}(g)E_{ij}$$

Résolution :

Soit $E_{kl} = e_k(e_l)^T$, où e_k et e_l sont les vecteurs de la base canonique.

$$\begin{aligned}
\tau_g(E_{kl}) &= \rho_g E_{kl} \phi_g^T \\
&= \rho_g (e_k(e_l)^T)\phi_g^T \\
&= (\rho_g e_k)((\phi_g e_l)^T) \\
&= \rho_{.,k}(g)\phi_{.,l}(g)^T \\
&= (\rho_{i,k}(g)\phi_{j,l}(g))_{i,j} \\
&= \sum_{i,j} \rho_{i,k}(g)\phi_{j,l}(g)E_{i,j}
\end{aligned}$$

Où :
$\rho_g e_k$ donne la k-ième colonne de ρ_g
$\phi_g e_l$ donne le l-ième colonne de ϕ_g
Le produit matriciel extérieur (outer) de ces deux vecteurs reconstruit une matrice qui peut s'écrire sur la base E_{ij} . \

Énoncé :

3.Prouver que $\chi_\tau(g) = \chi_\rho(g)\chi_\phi(g)$

Résolution :

Pour obtenir la trace, il faut calculer la somme des coefficients diagonaux. Pour $w : U \to U$, et U a pour base $\{u_k\}_{1 \leq k \leq mn}$, on a $w(u_i) = \sum_{k=1}^{mn} w_{k,i}u_k$ et la définition de la trace : $\mathrm{tr}(w) = \sum_{i=1}^{mn}(w(u_i))_i = \sum_{i=1}^{mn} w_{ii}$.

On peux définir une bijection linéaire f entre une base de U et celle de $M_{m,n}(\mathbb{C})$.

Considérons une matrice $A \in M_{m,n}(\mathbb{C})$:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Après linéarisation colonne par colonne, chaque élément a_{ij} est associé à un indice k donné par la formule :

$$k = (j-1)m + i.$$

La correspondance est représentée ainsi :

$$\begin{bmatrix} 1 & m+1 & \cdots & (n-1)m+1 \\ 2 & m+2 & \cdots & (n-1)m+2 \\ \vdots & \vdots & \ddots & \vdots \\ m & 2m & \cdots & nm \end{bmatrix}.$$

Grace à cette formule, on peut definir la bijection $\nu : \mathbb{N}^* \to \mathbb{N}^* \times \mathbb{N}^*$ tel que $\nu(k) = (i, j)$.

Exemple : pour $m = 3$ et $n = 4$, la matrice est donnée par :

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix}.$$

Les indices linéarisés sont :

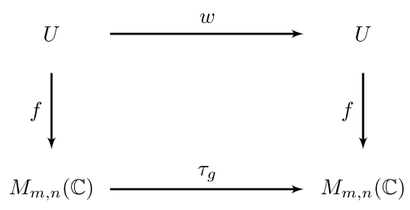
$$\begin{bmatrix} 1 & 4 & 7 & 10 \\ 2 & 5 & 8 & 11 \\ 3 & 6 & 9 & 12 \end{bmatrix}.$$

$f : U \to M_{m,n}(\mathbb{C})$ tel que $f(u_k) = E_{\nu(k)} = E_{i,j}$

In [37]:
```
dot_content = r"""digraph CommutativeDiagram {
    // Graph attributes
    rankdir=LR;
    bgcolor="transparent";

    // Graph-level positioning
    center=true;

    // Node attributes
    //node [style=filled, fillcolor=transparent shape=none, fontname="Arial"];

    // Nodes with precise positioning
    U1 [label="U", pos="0,1!", margin="0,0",shape=none];
    U2 [label="U", pos="2,1!", margin="0,0",shape=none];
    M1 [label="M_{m,n}(\\mathbb{C})", pos="0,0!", margin="0,0",shape=none];
    M2 [label="M_{m,n}(\\mathbb{C})", pos="2,0!", margin="0,0",shape=none];

    // Edges with dummy nodes
    //U1 -> f1 [style=invis];
    U1 -> U2 [style=solid];
```

```
    //U2 -> f2 [style=invis];
    M1 -> M2 [style=solid];

    //U1 -> w [style=invis];
    U1 -> M1 [style=solid];

    //M1 -> tau [style=invis];
    U2 -> M2 [style=solid];
    // Dummy label nodes
    f1 [label="f", pos="-0.1,0.5!", shape=none,margin="0,0"];
    f2 [label="f", pos="1.9,0.5!", shape=none];
    w [label="w", pos="1,1.1!", shape=none];
    tau [label="\\tau_g", pos="1,0.1!", shape=none];


}
"""


# Save the DOT content to a file
with open('commutative_diagram.dot', 'w') as f:
    f.write(dot_content)

# Define the path for the LaTeX file
tex_file_path = 'commutative_diagram'

# Convert the DOT file to LaTeX
subprocess.run(f'dot2tex --prog "neato" --docpreamble "\\usepackage[utf8]{{inputenc}} \\usepackage[T1]{{fontenc}} \\usepackage{{amssymb}}" -tmath --autosize "{tex_file_path}.d

# Insert LaTeX code into the notebook
insert_standalone(f"{tex_file_path}.tex")

# Compile the LaTeX file to generate the PDF
subprocess.run(f'xelatex "{tex_file_path}.tex"', shell=True)

# Convert the PDF to SVG
subprocess.run(f'pdf2svg "{tex_file_path}.pdf" "{tex_file_path}.svg"', shell=True)

# Display the SVG file in the notebook
SVG(f"{tex_file_path}.svg")
```

Out[37]:



$$w = f^{-1} \circ \tau_g \circ f$$
$$\mathrm{tr}(w) = \mathrm{tr}(f^{-1} \circ \tau_g \circ f)$$

Pour calculer la trace, commençons par calculer $w(u_p)$. On remplace $\rho_{i,k}(g)\phi_{j,l}(g)$ par $\alpha_{i,j}^{k,l}$ pour simplifier les calculs. k, l étant quelconque, on peut prendre $\nu(p) = (k, l)$:

$$
\begin{aligned}
w(u_p) &= f^{-1} \circ \tau_g \circ f(u_p) \\
&= f^{-1} \circ \tau_g \circ E_{\nu(p)} \\
&= f^{-1} \circ \tau_g(E_{k,l}) \\
&= f^{-1} \left(\sum_{i,j} \alpha_{i,j}^{k,l} E_{i,j} \right) \\
&= \sum_{i,j} \alpha_{i,j}^{k,l} f^{-1}(E_{i,j}) \\
&= \sum_{r} \alpha_{\nu(r)}^{k,l} u_r
\end{aligned}
$$

Prenons maintenant la p-ieme composante de $w(u_p)$:

$$
\begin{aligned}
\left(w(u_p)\right)_p &= \left(\sum_{r} \alpha_{\nu(r)}^{k,l} u_r \right)_p && (12)\\
&= \alpha_{\nu(p)}^{k,l} && (13)\\
&= \alpha_{k,l}^{k,l} && (14)\\
&= \rho_{k,k}(g)\phi_{l,l}(g) && (15)
\end{aligned}
$$

Finalement, calculons la trace :

$$
\begin{aligned}
\mathrm{tr}(w) &= \sum_{p=1}^{mn} \left(w(u_p)\right)_p && (16)\\
&= \sum_{k,l} \rho_{k,k}(g)\phi_{l,l}(g) && (17)\\
&= \sum_{k=1}^{m} \sum_{l=1}^{n} \rho_{k,k}(g)\phi_{l,l}(g) && (18)\\
&= \sum_{k=1}^{m} \rho_{k,k}(g) \sum_{l=1}^{n} \phi_{l,l}(g) && (19)\\
&= \chi_\phi(g)\chi_\rho(g) && (20)
\end{aligned}
$$

Finalement nous appliquons l'identification d'un endomorphisme avec une matrice :

$$
\begin{aligned}
w &\sim W \in M_{mn,mn}(\mathbb{C}) \\
f &\sim F \in M_{mn,mn}(\mathbb{C}) \\
\tau_g &\sim T_g \in M_{mn,mn}(\mathbb{C}) \\
tr(W) &= tr(F^{-1} T_g F) = tr(T_g)
\end{aligned}
$$

Donc $tr(w) = tr(\tau_g) = \chi_\tau(g)$.

Énoncé :

4.Soit G un groupe fini et χ, ψ deux caractères de G. Montrer que le produit point par point $\chi \cdot \psi : g \mapsto \chi(g)\psi(g)$ est un caractère de G .

Résolution :

Tout d'abord nous avons déjà démontré que $\forall g \in G, \quad g \mapsto \chi_\rho(g)\chi_\phi(g)$ est un caractère par construction de χ_τ.

Nous pouvons aussi le démontrer par l'étude de $\chi_\rho(g)\chi_\phi(g)$. Soit G un groupe fini, et χ, ψ deux caractères de G. Nous voulons montrer que leur produit point par point, défini par $(\chi\psi)(g) = \chi(g) \cdot \psi(g)$ pour tout $g \in G$, est également un caractère. Cela signifie qu'il s'agit d'un morphisme de groupe de G dans $\mathbb{C}^*$.\

Pour tout $g, h \in G$:

$$(\chi\psi)(gh) = \chi(gh) \cdot \psi(gh)$$

En utilisant le fait que χet ψ sont des morphismes de groupe, nous avons :

$$\chi(gh) = \chi(g)\chi(h) \quad \text{et} \quad \psi(gh) = \psi(g)\psi(h).$$

Substituons ces expressions dans $(\chi\psi)(gh)$:

$$(\chi\psi)(gh) = \chi(g)\chi(h) \cdot \psi(g)\psi(h).$$

Réorganisons les termes :

$$(\chi\psi)(gh) = \big(\chi(g)\psi(g)\big) \cdot \big(\chi(h)\psi(h)\big).$$

Par définition de $\chi\psi$, cela devient :

$$(\chi\psi)(gh) = (\chi\psi)(g) \cdot (\chi\psi)(h).$$

Ainsi, $\chi\psi$ vérifie la propriété de morphisme de groupe. Cela montre que le produit point par point de deux caractères χ et ψ est lui-même un caractère.

Coefficients de Clebsch-Gordan (ouverture - calculs faux à vérifier)

▶ calculs faux à vérifier