4. PRAKTIKA:
Grafoak

Grafo baten erpinak lotzen dituzten arkuak definitzeko honako notazio hau erabiliko dugu: (P, Q): P⟶Q. Arku guztiak zerrenda baten definituko dira.

Praktika honetan erabiliko ditugun Mathematica-ko funtzioak Combinatorica paketean daude eta bertan sartu beharko da:

Needs["Combinatorica`"]

GraphPlot[grafo, aukerak] Esandako grafoa adieraziko du azaldutako aukerekin. Aukeren artean honako hauek ditugu:
VertexLabeling→True Erpinak etiketatzen ditu.
DirectedEdges→True Grafoa zuzendua da.

RegularQ[g] True agertuko da g grafoa erregularra bada.

DeleteEdges[g, zerrenda] Zerrendan adierazitako g-ren arkuak ezabatzen ditu.

DeleteVertex[g, v] g grafoaren v erpina ezabatuko du.

DeleteVertices[g, zerrenda] Zerrendako g-ren erpinak ezabatuko ditu.

TreeQ[g] True agertuko da g grafoa zuhaitza bada

EulerianQ[g] True agertuko da g grafoa euleriarra bada.

RegularGraph[q, n] n erpineko q-erregularra den grafo bat lortuko du.

ExactRandomGraph[n, e] n erpineko eta e arku dituen zorizko grafo etiketatu bat lortuko da.

EulerianCycle[g] Existitzen bada, g grafoaren zirkuitu euleriarra aurkituko du.

Degrees[g] g grafoaren erpin bakoitzaren gradua lortuko du. Zuzendua den kasuan irteera graduak lortuko ditu.

FromAdjacencyMatrix[matrizea, baldintzak] grafo baten auzokidetasun matrizea definituko du. Pisuen bidez definitzerakoan **EdgeWeight** baldintza jarri beharko dugu pisuak onartzeko. Grafoa zuzendua den kasuan **Type→Directed** baldintza jarri beharko da.

MinimumSpanningTree[grafoa] Grafoaren zuhaitz estaltzaile minimala aurkituko du.

2. PRAKTIKA:

Logika matematikoa

1. Tautologiak, kontraesanak eta baliokidetasunak

TautologyQ[adierazpena, {a_1, a_2, ...}] a_1, a_2, … aldagaietan emandako adierazpena tautologia bat bada ikusten du.

Equivalent[e_1,e_2] $e_1 \Leftrightarrow e_2$ ($e_1 \equiv e_2$) baliokidetasun logikoa adieraziko du.

LogicalExpand[adierazpena] adierazpen logikoa garatu eta sinplifikatu egingo du.

Implies[p,q] $p \longrightarrow q$ inplikazio logikoa adieraziko du.

Simplify[adierazpena] Emandako adierazpena ahal den gehien sinplifikatuko du.

!**adierazpena** Not ($\neg$) funtzio logikoa adieraziko du.

3. PRAKTIKA:
Multzoak, aplikazioak, zatigarritasuna eta konbinatoria

1. Multzoak

Union[zerrenda1, zerrenda2, ...] zerrenda1, zerrenda2, … adierazpenetan agertzen diren elementu desberdin guztiak agertuko dira.

Intersection[zerrenda1, zerrenda2, ...] zerrenda1, zerrenda2, … adierazpenetan agertzen diren elementu komunak agertuko dira.

Complement[zerrenda1, zerrenda2] zerrenda2-n ez dauden zerrenda1-eko elementuak agertuko dira.

Subsets[zerrenda] *zerrenda*-ko azpimultzo posibleak agertuko dira.

Length[zerrenda] *zerrenda*-ko elementuen kopurua agertuko da.

2. Funtzioak

- **Parametrorik ez badago**

Solve[*sistema***,**{*ezezagunak*}**]** Sistemaren soluzioa lortuko du, existitzen bada.

NSolve[*sistema***,**{*ezezagunak*}**]** Sistemaren soluzio hurbildua lortuko du, existitzen bada.

- **Parametroak badaude:**

Reduce[*sistema,{ezezagunak}*] Sistemaren ebazpenerako parametroen arabera aukera guztiak aztertuko ditu.

If[baldintza, t, f] t erantzuna emango du baldintza egiaztatzen bada eta **f** emango du baldintza egiaztatzen ez bada.

3. Zatigarritasuna

GCD[n_1, n_2, ...] n_1, n_2, … balioen z.k.h. kalkulatuko du.

LCM[n_1, n_2, ...] n_1, n_2, … balioen m.k.t. kalkulatuko du

PrimeQ[adierazpena] True agertuko da adierazpena zenbaki lehena bada eta False konposatua bada.

CoprimeQ[n_1, n_2] True agertuko da n_1 eta n_2 elkar lehenak badira eta False aurkako kasuan.

NextPrime[n] n zenbakiaren hurbilen dagoen zenbaki lehena adieraziko du (n baino handiagoa).

FactorInteger[n] n zenbakiaren faktore lehenen zerrenda kalkulatu du, bakoitza bere berretzailearekin.

Divisors[n] n zenbakiaren zatitzaileen zerrenda kalkulatuko du.

4. Konbinatoria

Binomial[n, m] $\binom{n}{m}$ zenbaki binomikoa kalkulatuko du.