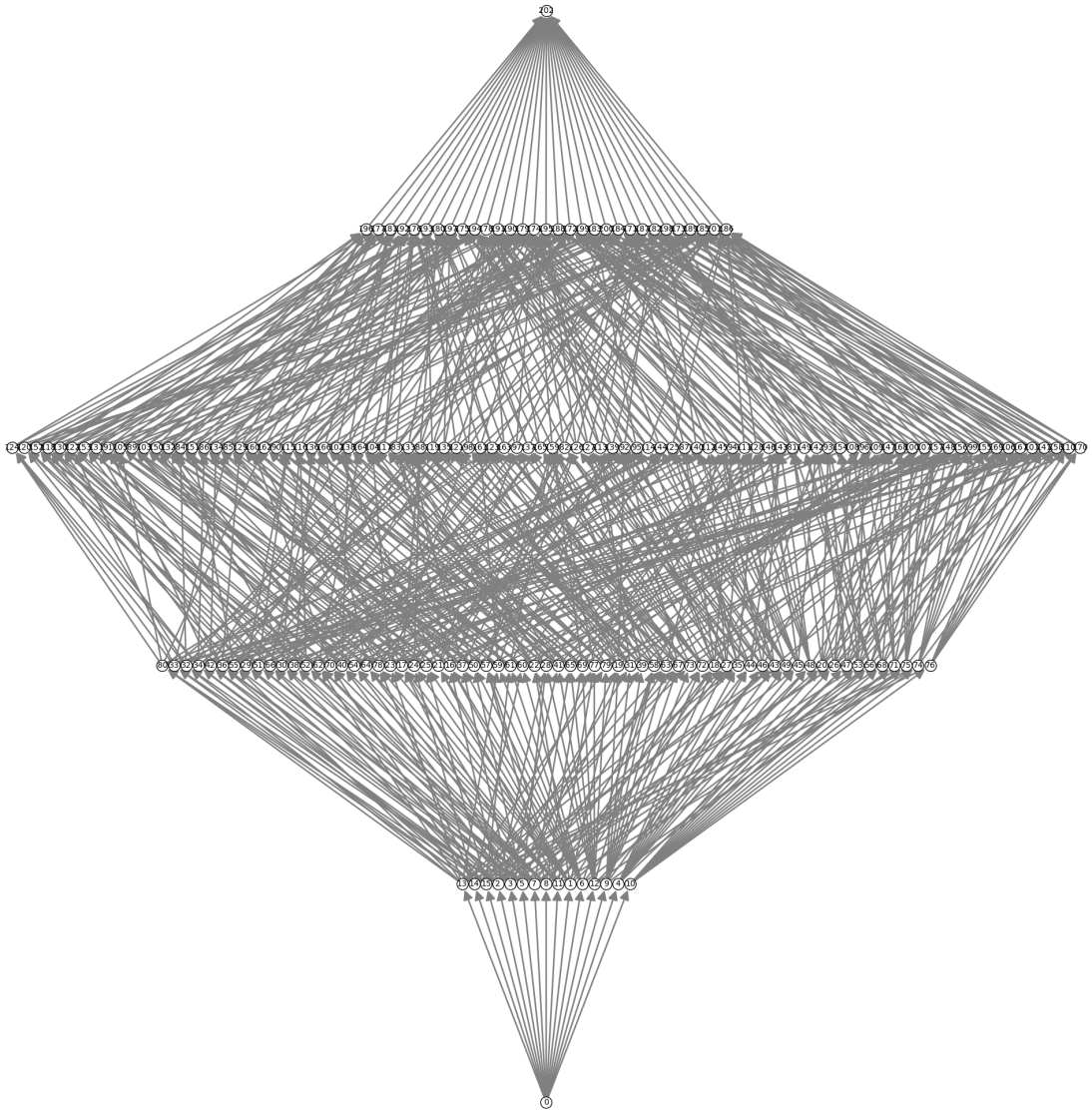
Chow Polynomial of Braid-6

First, we initialize the matroid Braid-6 and then obtain the lattice of flats. We relabel the flats for simple presentation.

[15]:
```
n = 6
edgelist = sorted(graphs.CompleteGraph(n).edges(labels=False))
matroid = Matroid(graph=edgelist, groundset=edgelist)
flats = [list(matroid.flats(i)) for i in range(n)]
all_elements = sum(flats, [])  # Flatten the list of flats
labels = {element: idx for idx, element in enumerate(all_elements)}
matroid.lattice_of_flats().plot(
    element_labels = labels, element_color = "white",
    figsize= (20,20) ,cover_color = "grey")
```

[15]:



We now generate the possible degrees that a flat can have in a monomial. For example, if $[x_1, x_2, x_3, x_4]$ is a chain and $[0, 1, 0, 2]$ is a weight, then we say that $x_2 x_4^2$ is an fy-monomial.

[16]:
```
def generate_weights(rank):
    weights = set()
    for i in range(1, rank):
        for j in range(rank):
            weight = [0] * rank

            if i >= j:
                weight[i] = j
                weights.add(tuple(weight))

            if rank - (i+1) > 1:
                # Recursion to get the complete list of weights...
                y = generate_weights(rank - (i + 1))
                for x in y:
                    temp_weight = weight.copy()
                    weights.add(tuple(temp_weight[:i+1] + x))

    return [list(w) for w in weights]

rank = matroid.rank()
weights = generate_weights(rank)
weights
```

[16]:
```
[[0, 0, 1, 0, 1],
 [0, 0, 0, 0, 2],
 [0, 1, 0, 0, 1],
 [0, 0, 1, 0, 0],
 [0, 0, 2, 0, 1],
 [0, 0, 0, 0, 1],
 [0, 0, 0, 3, 0],
 [0, 1, 0, 0, 0],
 [0, 0, 2, 0, 0],
 [0, 0, 0, 0, 4],
 [0, 1, 0, 1, 0],
 [0, 0, 0, 0, 0],
 [0, 0, 0, 1, 0],
 [0, 0, 0, 0, 3],
 [0, 0, 0, 2, 0],
 [0, 1, 0, 0, 2]]
```

Next, we compute all the possible fy-monomials in the example matroid.

[17]:
```
rflats = flats[1:] #empty flat is not required.
fy_monomials_list = [[] for _ in range(rank)]

def generate_monomials(weight, flats):
    if sum(weight) == 0: return [[]]
    # Find the first non-zero weight
    start_index = next(i for i in range(len(weight)) if weight[i] != 0)
    result = []

    for initial_flat in flats[start_index]:
        initial_monomial = [initial_flat] * weight[start_index]
        potential_combinations = [initial_monomial]

        for i in range(start_index + 1, len(weight)):
            if weight[i] != 0:
                new_combinations = []
                for flat in flats[i]:
                    for combo in potential_combinations:
                        if combo[-1].issubset(flat):
                            new_combo = combo + [flat] * weight[i]
                            new_combinations.append(new_combo)
                potential_combinations = new_combinations

        result.extend(potential_combinations)

    return result

for weight in weights:
    fy_monomials_list[sum(weight)].extend(generate_monomials(weight, rflats))

#example fy-monomial
print(fy_monomials_list[2][0])
```

```
[frozenset({(0, 1), (2, 3), (0, 2), (1, 2), (0, 3), (1, 3)}), frozenset({(0, 1),
(2, 4), (1, 2), (0, 4), (3, 4), (1, 5), (0, 3), (1, 4), (2, 3), (0, 2), (4, 5),
(0, 5), (2, 5), (1, 3), (3, 5)})]
```

We write some simplification functions to make fy-monomials look simpler. Then we write out all fy-monomials of braid-5 in this simple form.

[18]:
```
def simplify(monomial):
    return tuple(sorted([labels[x] for x in monomial]))

def set_simplify(monomial_set):
    return set(sorted([simplify(x) for x in monomial_set]))
```

The symmetric group $G = S_6$ acts on the example matroid. We now set up the appropriate functions required to compute the actions on the vertex set and obtain the set of orbits and their stabilizer

groups.

[19]:
```
G = SymmetricGroup(range(n))

def action_on_flats(g, m):
    def action_on_groundset(g, x):
        return tuple(sorted(g(y) for y in x))
    return frozenset(sorted([action_on_groundset(g,x) for x in m]))

def action_on_braidfymonomials(g, monomial):
    return tuple(sorted([action_on_flats(g,m) for m in monomial]))

def stab(G, m, action):
    return G.subgroup(set(g for g in G if action(g, m) == tuple(m)))

def orbit(G, m, action):
    return frozenset(sorted(action(g, m) for g in G))

def orbits(G, X, action):
    return set(orbit(G, x, action) for x in X)
```

Finally, we compute the orbits of the fy-monomials under the action of G.

[20]:
```
fy_monomials_orbits = [orbits(G, fy_monomials_list[i],␣
 ↪action_on_braidfymonomials)
    for i in range(rank)
    ]

orbits_dict = {}
for idx, orbit_set in enumerate(fy_monomials_orbits):
    print(f"\nrank: {idx}")
    for x in orbit_set:
        orbits_dict[x] = len(orbits_dict)
        print(f"{orbits_dict[x]}: {set_simplify(x)}\n")
```

```
rank: 0
0: {()}


rank: 1
1: {(176,), (182,), (185,), (175,), (188,), (194,), (197,), (187,), (200,),
(190,), (180,), (193,), (199,), (173,), (179,)}

2: {(202,)}

3: {(118,), (121,), (127,), (124,), (130,), (133,), (139,), (142,), (145,),
(84,), (90,), (148,), (93,), (99,), (160,), (96,), (102,), (108,), (166,),
```

(105,), (169,), (114,), (117,), (126,), (132,), (129,), (135,), (138,), (89,),
(141,), (86,), (150,), (95,), (98,), (165,), (162,), (168,), (113,), (110,),
(116,), (119,), (128,), (131,), (140,), (85,), (143,), (149,), (94,), (152,),
(91,), (97,), (158,), (155,), (161,), (100,), (164,), (109,), (115,), (167,),
(112,)}

4: {(186,), (172,), (201,), (178,), (171,), (174,)}

5: {(57,), (63,), (75,), (17,), (26,), (35,), (56,), (62,), (16,), (74,), (80,),
(19,), (77,), (28,), (43,), (58,), (67,), (18,), (27,), (36,)}

6: {(192,), (198,), (195,), (191,), (184,), (181,), (177,), (196,), (183,),
(189,)}

7: {(51,), (60,), (66,), (69,), (72,), (78,), (23,), (20,), (29,), (32,), (38,),
(44,), (41,), (47,), (50,), (53,), (59,), (68,), (65,), (71,), (25,), (22,),
(31,), (37,), (34,), (40,), (49,), (46,), (52,), (55,), (61,), (64,), (70,),
(73,), (76,), (21,), (79,), (24,), (30,), (33,), (42,), (39,), (45,), (48,),
(54,)}

8: {(147,), (153,), (137,), (101,), (146,), (120,), (104,), (154,), (123,),
(136,), (151,), (103,), (144,), (122,), (134,)}

9: {(163,), (92,), (82,), (111,), (156,), (170,), (159,), (88,), (107,), (81,),
(87,), (106,), (125,), (83,), (157,)}

rank: 2
10: {(31, 191), (70, 183), (78, 196), (50, 198), (61, 198), (30, 192), (33,
191), (76, 189), (71, 184), (76, 198), (20, 184), (22, 181), (40, 196), (47,
183), (66, 181), (51, 192), (49, 183), (73, 195), (78, 191), (33, 177), (70,
196), (42, 189), (68, 189), (29, 191), (23, 184), (61, 177), (37, 195), (30,
189), (41, 195), (32, 192), (46, 191), (23, 177), (38, 196), (72, 191), (55,
184), (21, 181), (47, 198), (25, 177), (51, 198), (38, 189), (72, 184), (22,
183), (53, 198), (65, 181), (41, 181), (24, 183), (66, 198), (44, 195), (49,
191), (79, 192), (21, 184), (48, 195), (59, 177), (64, 191), (60, 196), (52,
192), (53, 184), (34, 189), (45, 198), (69, 183), (69, 192), (20, 183), (45,
191), (37, 196), (52, 196), (48, 183), (46, 195), (39, 189), (54, 196), (32,
177), (59, 195), (25, 181), (65, 192), (73, 184), (64, 181), (40, 181), (34,
177), (60, 177), (42, 181), (71, 189), (29, 192), (31, 189), (55, 192), (24,
177), (50, 196), (44, 198), (68, 183), (79, 195), (54, 184), (39, 195)}

11: {(187, 187), (200, 200), (180, 180), (194, 194), (197, 197), (190, 190),
(179, 179), (199, 199), (176, 176), (173, 173), (193, 193), (182, 182), (175,
175), (188, 188), (185, 185)}

12: {(21, 202), (54, 202), (23, 202), (34, 202), (45, 202), (78, 202), (60,
202), (47, 202), (69, 202), (51, 202), (25, 202), (38, 202), (49, 202), (71,

202), (29, 202), (40, 202), (73, 202), (55, 202), (20, 202), (31, 202), (53, 202), (64, 202), (22, 202), (42, 202), (33, 202), (44, 202), (66, 202), (24, 202), (46, 202), (68, 202), (79, 202), (72, 202), (48, 202), (39, 202), (37, 202), (59, 202), (70, 202), (50, 202), (61, 202), (30, 202), (41, 202), (52, 202), (65, 202), (76, 202), (32, 202)}

13: {(36, 196), (28, 192), (43, 198), (77, 181), (80, 177), (67, 189), (56, 198), (58, 195), (16, 177), (57, 196), (18, 183), (63, 191), (62, 192), (17, 181), (19, 184), (27, 191), (74, 184), (75, 183), (26, 189), (35, 195)}

14: {(87, 87), (107, 107), (125, 125), (106, 106), (156, 156), (81, 81), (163, 163), (83, 83), (92, 92), (170, 170), (157, 157), (82, 82), (159, 159), (111, 111), (88, 88)}

15: {(164, 202), (142, 202), (100, 202), (133, 202), (109, 202), (155, 202), (166, 202), (91, 202), (89, 202), (102, 202), (124, 202), (135, 202), (93, 202), (126, 202), (148, 202), (161, 202), (168, 202), (113, 202), (160, 202), (95, 202), (117, 202), (115, 202), (128, 202), (139, 202), (86, 202), (97, 202), (84, 202), (119, 202), (141, 202), (150, 202), (110, 202), (143, 202), (152, 202), (121, 202), (108, 202), (130, 202), (132, 202), (90, 202), (112, 202), (99, 202), (145, 202), (165, 202), (167, 202), (114, 202), (169, 202), (158, 202), (116, 202), (127, 202), (138, 202), (149, 202), (85, 202), (96, 202), (94, 202), (105, 202), (118, 202), (129, 202), (98, 202), (131, 202), (140, 202), (162, 202)}

16: {(146, 202), (134, 202), (137, 202), (153, 202), (154, 202), (144, 202), (120, 202), (104, 202), (101, 202), (123, 202), (151, 202), (103, 202), (136, 202), (147, 202), (122, 202)}

17: {(22, 176), (54, 190), (64, 200), (34, 190), (47, 187), (55, 194), (42, 194), (46, 182), (60, 199), (33, 188), (79, 185), (68, 200), (40, 187), (73, 193), (37, 173), (71, 199), (72, 197), (49, 194), (50, 176), (44, 175), (52, 182), (51, 180), (59, 200), (41, 188), (61, 197), (66, 193), (76, 185), (23, 179), (45, 179), (48, 190), (78, 185), (30, 175), (65, 199), (70, 193), (69, 197), (25, 182), (24, 180), (32, 187), (29, 173), (38, 179), (21, 175), (20, 173), (39, 180), (31, 176), (53, 188)}

18: {(202, 202)}

19: {(123, 123), (136, 136), (134, 134), (154, 154), (103, 103), (151, 151), (144, 144), (153, 153), (137, 137), (146, 146), (147, 147), (122, 122), (120, 120), (104, 104), (101, 101)}

20: {(163, 202), (159, 202), (170, 202), (92, 202), (87, 202), (82, 202), (106, 202), (88, 202), (83, 202), (107, 202), (81, 202), (111, 202), (125, 202), (156, 202), (157, 202)}

21: {(178, 178), (171, 171), (174, 174), (201, 201), (186, 186), (172, 172)}

22: {(198, 198), (196, 196), (177, 177), (181, 181), (183, 183), (192, 192), (189, 189), (191, 191), (195, 195), (184, 184)}

23: {(98, 98), (162, 162), (139, 139), (84, 84), (116, 116), (126, 126), (112, 112), (167, 167), (89, 89), (135, 135), (121, 121), (130, 130), (131, 131), (140, 140), (117, 117), (85, 85), (94, 94), (149, 149), (158, 158), (145, 145), (168, 168), (90, 90), (99, 99), (108, 108), (164, 164), (86, 86), (141, 141), (150, 150), (118, 118), (127, 127), (95, 95), (113, 113), (114, 114), (169, 169), (91, 91), (155, 155), (132, 132), (100, 100), (109, 109), (110, 110), (119, 119), (160, 160), (96, 96), (105, 105), (128, 128), (124, 124), (165, 165), (133, 133), (142, 142), (143, 143), (152, 152), (129, 129), (138, 138), (97, 97), (115, 115), (161, 161), (93, 93), (148, 148), (166, 166), (102, 102)}

24: {(34, 193), (47, 199), (79, 179), (49, 199), (31, 193), (78, 180), (59, 176), (61, 173), (64, 190), (41, 179), (33, 175), (59, 194), (76, 173), (76, 182), (66, 190), (46, 187), (40, 180), (72, 187), (21, 185), (69, 179), (69, 188), (54, 200), (25, 173), (72, 180), (78, 175), (42, 173), (45, 190), (55, 200), (45, 199), (68, 182), (48, 179), (21, 180), (39, 197), (37, 197), (65, 179), (32, 176), (52, 188), (50, 200), (41, 197), (33, 193), (64, 180), (70, 175), (47, 182), (30, 193), (60, 194), (51, 200), (73, 176), (53, 182), (22, 185), (53, 200), (66, 173), (49, 175), (79, 176), (68, 188), (24, 185), (73, 187), (29, 193), (37, 194), (44, 199), (34, 173), (39, 187), (48, 199), (32, 193), (65, 190), (55, 176), (20, 185), (40, 197), (60, 175), (51, 190), (42, 197), (44, 194), (71, 187), (54, 180), (29, 190), (31, 187), (70, 188), (24, 175), (50, 194), (61, 194), (30, 188), (52, 200), (46, 199), (23, 176), (23, 185), (25, 185), (71, 182), (38, 188), (20, 182), (22, 179), (38, 197)}

25: {(63, 202), (80, 202), (56, 202), (67, 202), (36, 202), (58, 202), (27, 202), (16, 202), (18, 202), (62, 202), (77, 202), (75, 202), (35, 202), (57, 202), (26, 202), (17, 202), (28, 202), (19, 202), (74, 202), (43, 202)}

26: {(60, 172), (72, 201), (30, 174), (41, 186), (59, 201), (70, 201), (25, 178), (22, 172), (21, 174), (23, 171), (55, 178), (34, 186), (42, 178), (24, 172), (46, 178), (39, 172), (29, 171), (40, 171), (22, 174), (70, 178), (41, 172), (64, 201), (68, 171), (48, 186), (66, 201), (59, 171), (52, 174), (30, 171), (51, 178), (71, 172), (76, 186), (38, 178), (49, 178), (31, 172), (20, 172), (21, 171), (69, 174), (34, 174), (47, 171), (78, 186), (54, 186), (33, 172), (66, 178), (38, 171), (60, 201), (31, 174), (40, 186), (64, 171), (50, 172), (37, 172), (55, 186), (73, 201), (65, 172), (48, 174), (33, 186), (79, 201), (45, 178), (61, 174), (32, 171), (29, 172), (73, 178), (54, 174), (45, 171), (47, 186), (69, 201), (24, 178), (39, 178), (53, 186), (71, 201), (52, 178), (46, 174), (44, 174), (42, 186), (72, 174), (37, 171), (51, 172), (50, 174), (32, 186), (61, 201), (53, 172), (65, 201), (76, 201), (78, 201), (25, 174), (20, 171), (49, 186), (44, 171), (79, 186), (68, 201), (23, 178)}

27: {(62, 200), (63, 193), (63, 199), (27, 175), (67, 193), (77, 185), (17, 180), (77, 197), (80, 193), (19, 180), (27, 187), (56, 199), (27, 190), (35,

179), (58, 199), (75, 185), (16, 175), (26, 173), (75, 188), (18, 175), (35,
194), (63, 180), (57, 200), (26, 188), (57, 197), (62, 193), (17, 173), (36,
180), (77, 190), (28, 176), (17, 179), (19, 176), (19, 182), (28, 188), (43,
182), (57, 175), (74, 185), (74, 200), (35, 187), (43, 194), (75, 199), (26,
187), (56, 173), (67, 182), (58, 176), (56, 200), (80, 185), (36, 188), (36,
194), (67, 197), (80, 194), (28, 190), (58, 197), (74, 187), (16, 173), (16,
176), (18, 179), (43, 190), (62, 179), (18, 182)}

28: {(67, 178), (17, 171), (36, 178), (80, 178), (28, 174), (74, 171), (43,
174), (19, 174), (28, 186), (74, 186), (43, 186), (16, 172), (74, 201), (18,
178), (62, 178), (67, 174), (56, 171), (58, 174), (27, 174), (63, 201), (27,
171), (36, 186), (75, 172), (80, 186), (27, 186), (56, 201), (67, 201), (26,
172), (58, 201), (16, 171), (35, 178), (80, 201), (16, 174), (18, 171), (62,
171), (18, 174), (77, 174), (17, 172), (28, 172), (77, 186), (19, 172), (62,
201), (17, 178), (19, 178), (75, 186), (57, 174), (77, 201), (35, 171), (43,
178), (57, 171), (26, 171), (35, 186), (63, 172), (75, 201), (26, 186), (63,
178), (56, 172), (57, 201), (36, 172), (58, 172)}

rank: 3
29: {(195, 195, 195), (198, 198, 198), (196, 196, 196), (189, 189, 189), (184,
184, 184), (177, 177, 177), (191, 191, 191), (183, 183, 183), (192, 192, 192),
(181, 181, 181)}

30: {(82, 82, 202), (163, 163, 202), (81, 81, 202), (107, 107, 202), (125, 125,
202), (88, 88, 202), (92, 92, 202), (159, 159, 202), (111, 111, 202), (157, 157,
202), (156, 156, 202), (83, 83, 202), (87, 87, 202), (106, 106, 202), (170, 170,
202)}

31: {(98, 98, 202), (124, 124, 202), (143, 143, 202), (105, 105, 202), (85, 85,
202), (155, 155, 202), (133, 133, 202), (89, 89, 202), (115, 115, 202), (160,
160, 202), (84, 84, 202), (129, 129, 202), (148, 148, 202), (126, 126, 202),
(152, 152, 202), (90, 90, 202), (135, 135, 202), (116, 116, 202), (94, 94, 202),
(142, 142, 202), (161, 161, 202), (139, 139, 202), (165, 165, 202), (86, 86,
202), (112, 112, 202), (131, 131, 202), (95, 95, 202), (138, 138, 202), (118,
118, 202), (121, 121, 202), (99, 99, 202), (110, 110, 202), (91, 91, 202), (117,
117, 202), (162, 162, 202), (114, 114, 202), (140, 140, 202), (166, 166, 202),
(130, 130, 202), (108, 108, 202), (168, 168, 202), (149, 149, 202), (127, 127,
202), (100, 100, 202), (141, 141, 202), (97, 97, 202), (164, 164, 202), (167,
167, 202), (145, 145, 202), (96, 96, 202), (93, 93, 202), (119, 119, 202), (158,
158, 202), (102, 102, 202), (169, 169, 202), (109, 109, 202), (150, 150, 202),
(128, 128, 202), (132, 132, 202), (113, 113, 202)}

32: {(69, 202, 202), (47, 202, 202), (29, 202, 202), (41, 202, 202), (31, 202,
202), (53, 202, 202), (70, 202, 202), (42, 202, 202), (54, 202, 202), (66, 202,
202), (78, 202, 202), (44, 202, 202), (38, 202, 202), (50, 202, 202), (65, 202,
202), (55, 202, 202), (39, 202, 202), (45, 202, 202), (68, 202, 202), (52, 202,
202), (30, 202, 202), (34, 202, 202), (40, 202, 202), (24, 202, 202), (46, 202,

```
202), (51, 202, 202), (25, 202, 202), (64, 202, 202), (48, 202, 202), (20, 202,
202), (59, 202, 202), (71, 202, 202), (37, 202, 202), (49, 202, 202), (61, 202,
202), (76, 202, 202), (21, 202, 202), (32, 202, 202), (22, 202, 202), (72, 202,
202), (73, 202, 202), (79, 202, 202), (33, 202, 202), (60, 202, 202), (23, 202,
202)}

33: {(194, 194, 194), (179, 179, 179), (187, 187, 187), (175, 175, 175), (200,
200, 200), (193, 193, 193), (180, 180, 180), (182, 182, 182), (176, 176, 176),
(185, 185, 185), (188, 188, 188), (173, 173, 173), (190, 190, 190), (197, 197,
197), (199, 199, 199)}

34: {(174, 174, 174), (201, 201, 201), (172, 172, 172), (178, 178, 178), (171,
171, 171), (186, 186, 186)}

35: {(202, 202, 202)}

36: {(19, 202, 202), (43, 202, 202), (56, 202, 202), (77, 202, 202), (16, 202,
202), (28, 202, 202), (27, 202, 202), (63, 202, 202), (74, 202, 202), (62, 202,
202), (18, 202, 202), (75, 202, 202), (80, 202, 202), (58, 202, 202), (36, 202,
202), (26, 202, 202), (57, 202, 202), (35, 202, 202), (67, 202, 202), (17, 202,
202)}

37: {(104, 104, 202), (123, 123, 202), (120, 120, 202), (101, 101, 202), (136,
136, 202), (147, 147, 202), (154, 154, 202), (146, 146, 202), (153, 153, 202),
(137, 137, 202), (151, 151, 202), (122, 122, 202), (103, 103, 202), (134, 134,
202), (144, 144, 202)}


rank: 4
38: {(202, 202, 202, 202)}
```

Now, we compute the stabilizer of each orbit fy_monomials_orbits(rank) for all ranks.

[21]:
```python
for orbits in fy_monomials_orbits:
    fn = ClassFunction(G, [0] * len(G.conjugacy_classes()))
    orbits_simplify = {orbits_dict[orbit] for orbit in orbits}

    for orbit in orbits:
        orbit_num = orbits_dict[orbit]
        orbit_stab = stab(G, tuple(orbit)[0], action_on_braidfymonomials)
        print(f"Orbit {orbit_num} is stabilized by a subgroup of order␣
↪{orbit_stab.order()}")
        fn += orbit_stab.trivial_character().induct(G)

    print(f"\nThe permutation representation for {orbits_simplify} is")
    for x, y in fn.decompose():
        print(f"{x} copies of {list(y.values())}")
```

```
    print("\n")
```

```
Orbit 0 is stabilized by a subgroup of order 720

The permutation representation for {0} is
1 copies of [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]


Orbit 1 is stabilized by a subgroup of order 48
Orbit 2 is stabilized by a subgroup of order 720
Orbit 3 is stabilized by a subgroup of order 12
Orbit 4 is stabilized by a subgroup of order 120
Orbit 5 is stabilized by a subgroup of order 36
Orbit 6 is stabilized by a subgroup of order 72
Orbit 7 is stabilized by a subgroup of order 16
Orbit 8 is stabilized by a subgroup of order 48
Orbit 9 is stabilized by a subgroup of order 48

The permutation representation for {1, 2, 3, 4, 5, 6, 7, 8, 9} is
9 copies of [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
2 copies of [5, -1, 1, 3, -1, -1, 2, 1, -1, 0, 0]
2 copies of [5, 1, 1, -3, -1, 1, 2, -1, -1, 0, 0]
7 copies of [5, 3, 1, -1, 2, 0, -1, 1, -1, 0, -1]
9 copies of [9, 3, 1, 3, 0, 0, 0, -1, 1, -1, 0]
1 copies of [10, 2, -2, -2, 1, -1, 1, 0, 0, 0, 1]
2 copies of [16, 0, 0, 0, -2, 0, -2, 0, 0, 1, 0]


Orbit 10 is stabilized by a subgroup of order 8
Orbit 11 is stabilized by a subgroup of order 48
Orbit 12 is stabilized by a subgroup of order 16
Orbit 13 is stabilized by a subgroup of order 36
Orbit 14 is stabilized by a subgroup of order 48
Orbit 15 is stabilized by a subgroup of order 12
Orbit 16 is stabilized by a subgroup of order 48
Orbit 17 is stabilized by a subgroup of order 16
Orbit 18 is stabilized by a subgroup of order 720
Orbit 19 is stabilized by a subgroup of order 48
Orbit 20 is stabilized by a subgroup of order 48
Orbit 21 is stabilized by a subgroup of order 120
Orbit 22 is stabilized by a subgroup of order 72
Orbit 23 is stabilized by a subgroup of order 12
Orbit 24 is stabilized by a subgroup of order 8
Orbit 25 is stabilized by a subgroup of order 36
Orbit 26 is stabilized by a subgroup of order 8
Orbit 27 is stabilized by a subgroup of order 12
Orbit 28 is stabilized by a subgroup of order 12
```

```
The permutation representation for {10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 28} is
19 copies of [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
7 copies of [5, -1, 1, 3, -1, -1, 2, 1, -1, 0, 0]
8 copies of [5, 1, 1, -3, -1, 1, 2, -1, -1, 0, 0]
21 copies of [5, 3, 1, -1, 2, 0, -1, 1, -1, 0, -1]
1 copies of [9, -3, 1, -3, 0, 0, 0, 1, 1, -1, 0]
28 copies of [9, 3, 1, 3, 0, 0, 0, -1, 1, -1, 0]
1 copies of [10, -2, -2, 2, 1, 1, 1, 0, 0, 0, -1]
7 copies of [10, 2, -2, -2, 1, -1, 1, 0, 0, 0, 1]
12 copies of [16, 0, 0, 0, -2, 0, -2, 0, 0, 1, 0]


Orbit 29 is stabilized by a subgroup of order 72
Orbit 30 is stabilized by a subgroup of order 48
Orbit 31 is stabilized by a subgroup of order 12
Orbit 32 is stabilized by a subgroup of order 16
Orbit 33 is stabilized by a subgroup of order 48
Orbit 34 is stabilized by a subgroup of order 120
Orbit 35 is stabilized by a subgroup of order 720
Orbit 36 is stabilized by a subgroup of order 36
Orbit 37 is stabilized by a subgroup of order 48

The permutation representation for {32, 33, 34, 35, 36, 37, 29, 30, 31} is
9 copies of [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
2 copies of [5, -1, 1, 3, -1, -1, 2, 1, -1, 0, 0]
2 copies of [5, 1, 1, -3, -1, 1, 2, -1, -1, 0, 0]
7 copies of [5, 3, 1, -1, 2, 0, -1, 1, -1, 0, -1]
9 copies of [9, 3, 1, 3, 0, 0, 0, -1, 1, -1, 0]
1 copies of [10, 2, -2, -2, 1, -1, 1, 0, 0, 0, 1]
2 copies of [16, 0, 0, 0, -2, 0, -2, 0, 0, 1, 0]


Orbit 38 is stabilized by a subgroup of order 720

The permutation representation for {38} is
1 copies of [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
```