1). בקוד של החלק הראשון, ייבאנו שתי ספריות: iterations ו networkx. שתי ספריות אלה שימשו אותנו במהלך כתיבת הקוד ובהמשך נסביר כיצד. בקוד זה בעצם יצרנו את הפונקציה generate\_subgraphs שמקבלת כקלט מספר n שלם(הנחנו שהוא גדול מ0). פונקציה זאת מחזירה רשימה של כל התתי-גרף שעומדים בדרישות התרגיל(תתי גרפים קשירים שכוללים את כל הקודקודים שממוספרים מ1 עד n וכל גרף אינו איזומורפי לקודמו! למשל אם הגרף של 1 2 מופיע עבור n=2 אזי 2 1 לא יופיע גם כן-> היות והם איזומורפיים).

בפונקציה זו בעצם איתחלנו את המשתנה edges ע"י itertools.permutations מה שזה עושה בעצם זה יוצר כל חיבורים אפשריים של קשתות בין הקודקודים(קשתות מכוונות) , וכמו כן מייצרים את הרשימה subgraphs שאותה נחזיר מהפונקציה בסופו של דבר. לאחר מכן, בלופ שנע מ1 עד מספר הקשתות אנחנו נכנסים ללואה פנימית שנמתחשבת בכל האופציות של מספר קשתות r.

בשלב זה אנחנו יוצרים כל פעם גרף באמצעות nx.DiGraph(list(subset)) וגרף זה הינו אופציה להוספה בהמשך. כעת, בודקים באמת שהגרף מכיל n קודקודים, וגם כן שהינו קשיר! באמצעות הספרייה של networkx. בהנחה ועברנו את הבדיקות הנ"ל נבדוק אם הגרף איזומורפי לאיזשהו גרף אחר ברשימה הטמפורית שלנו, ומוסיפים רק אם איננו. ולאחר הלולאה מחזירים תשובה!

לאחר מכן מדפיסים הכל לפי הפורמט המבוקש וע"י שינוי n בקוד, נוכל לקבל תשובה עבור כל n שנרצה.

זמני הריצה גדלים אספוננציאלית בn ועל כן הפערים הגדולים בזמנים(היות ורצים על כל הקומבינציות של הקשתות).