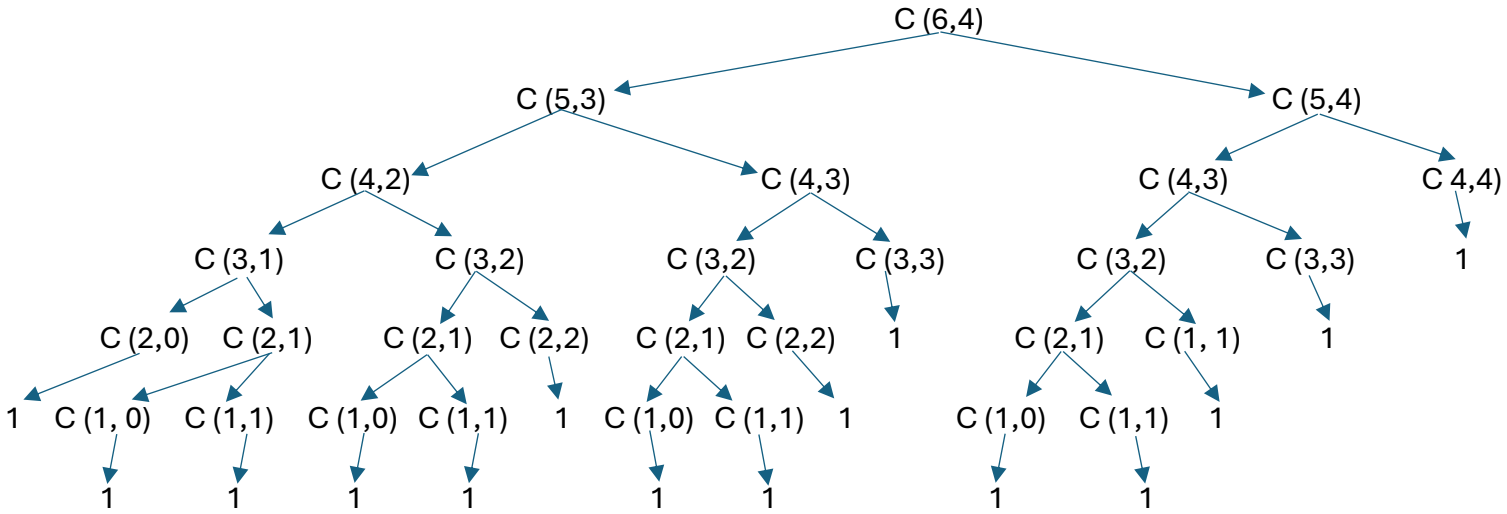
Oppgave 2 - Kalltre



Oppgave 4:

For større verdien av n i oppgave 3, ser man at programmet bruker utrolig lang tid ved større verdier. Grunnen til dette kan man se i kalltreet i oppgave 2. Man ser der at samme Fibonaccitall beregnes ut flere ganger, og man får dermed mye redundant kode og itereringer. Fibonaccitall er et typisk eksempel på hvor en iterativ tilnærming er bedre.

Oppgave 7:

Sammenliknet med den rekursive metoden i oppgave 3, er oppgave 6 (iterativ) mye bedre her. Dette kommer av hvordan tallene beregnes. Forskjellen er at metoden i oppgave 3 beregner samme verdi flere ganger, uten å lagre den, som fører til den beregner alle verdiene på nytt, for hver nye posisjon i trekanten. Den iterative metoden derimot, gjør ikke dette, som fører til at den er mye mer effektiv.

For å sammenlikne, kunne oppgave 3 bruke flere sekunder på n=30, mens oppgave 6 må først opp på n=200+. Det er likevel lett å se forskjellen på kjøretiden, ettersom den iterative ikke egentlig blir tregere (som den rekursive), men bruker bare lengere tid, ettersom det er flere tall å gå gjennom.