Datastructuren en Algoritmiek

Door: Jeroen Elschot en Rutger van Sinderen (EIN2b)

Documentatie: opdracht 4 (PERT-netwerk)

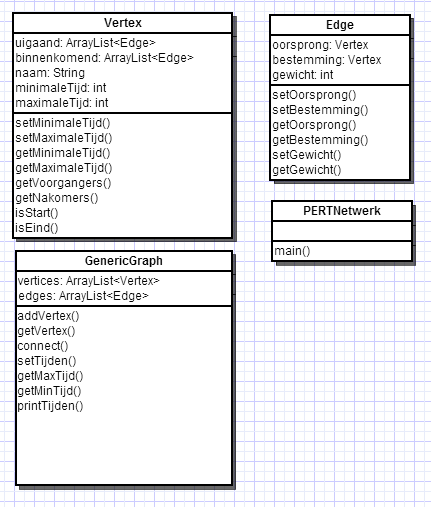
# Opdracht beschrijving

Voor opdracht 4 moesten we een datastructuur bedenken voor een PERT netwerk. Dit PERT netwerk moest vervolgens de kortste en langste voltooiingstijden kunnen berekenen voor de knopen in het netwerk. Voordat we konden beginnen moesten we eerst zelf oefenen met het uitrekenen van deze voltooiingstijden, hierna konden we methoden bedenken voor de programmatuur.

## Beschrijving uiteindelijk programma

Uiteindelijk hebben we een programma gemaakt waarmee PERT netwerken kunnen worden gemaakt. Om zo’n netwerk te maken moeten eerst de knopen gemaakt worden, vervolgens moeten de knopen toegevoegd worden naar het netwerk. Hierna moeten de knopen verbonden worden met de zijden.

# Datastructuur

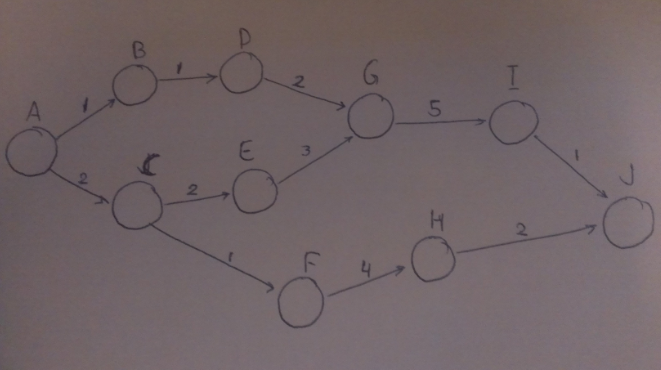


Hierboven is ons ontwerp van de datastructuur te zien. We zijn begonnen met het ontwerpen van de Vertex en de Edge klassen en de manier waarop deze twee moeten verbinden. Uiteindelijk hebben we ervoor gekozen dat de Vertex de uitgaande en binnenkomende edges opslaat in ArrayLists. De edges houden ook bij wat de oorsprong en bestemming is. Verder heeft een edge nog een gewicht. De Vertex weet van zichzelf bij het aanmaken van het netwerk nog niet de minimale en maximale voltooiingstijden. Deze worden uitgerekend door setMinimaleTijd() en setMaximaleTijd(). Deze waardes zijn vervolgens op te halen met de getters. De booleans isStart en isEind geven aan of een knoop een startknoop of een eindknoop is. Dat wil zeggen: of de knoop voorgangers of nakomers heeft of niet.

# Uitvoer

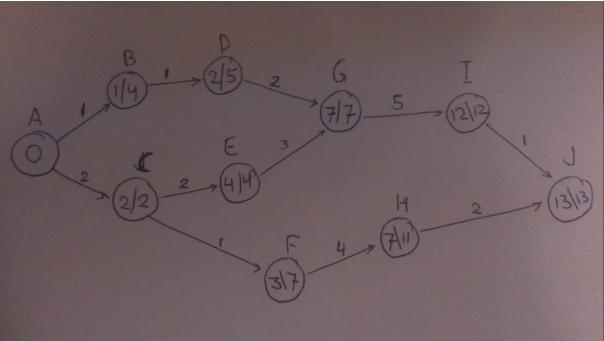
Om te kijken of de gemaakt programmatuur goed werkt hebben we zelf een PERT netwerk bedacht en daar de minimale en maximale voltooiingstijden van berekend.

Het betreffende PERT-netwerk:



Hiervan hebben we eerst zelf de voltooiingstijden per knoop uitgerekend:

Dit geeft onderstaand resultaat



De verwachte uitvoer

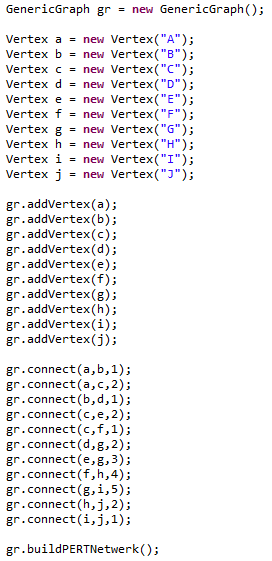
In de tabel hieronder valt te zien wat wij verwachten dat de uitvoer moet zijn

|  |  |  |
| --- | --- | --- |
| Knoop | Minimale tijd | Maximale tijd |
| A | 0 | 0 |
| B | 1 | 4 |
| C | 2 | 2 |
| D | 2 | 5 |
| E | 4 | 4 |
| F | 3 | 7 |
| G | 7 | 7 |
| H | 7 | 11 |
| I | 12 | 12 |
| J | 13 | 13 |

Nogmaals een overzicht van het PERT-Netwerk waaruit wij bovenstaande uitvoer verwachten.



# PERT-Netwerk construeren in het programma

Hier valt te zien hoe de PERT wordt ingevoerd.

Eerst wordt er een nieuwe GenericGraph gedeclareerd en geinitialiseerd.

Vervolgens worden alle knopen aangemaakt.

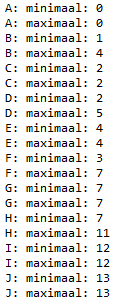
Vervolgens worden de knopen toegevoegd aan het netwerk met de addVertex methode.

Hierop worden de knopen met elkaar verbonden met de egdes via de connect methode.

En tot slot wordt met de methode buikdPERTNetwerk de minimale en maximale tijd berekend van elk knooppunt/Vertex.

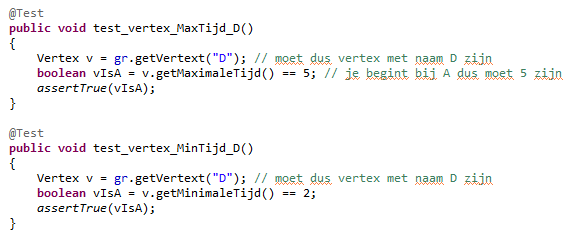
# De Uitvoer

|  |  |  |
| --- | --- | --- |
| Knoop | Minimale tijd | Maximale tijd |
| A | 0 | 0 |
| B | 1 | 4 |
| C | 2 | 2 |
| D | 2 | 5 |
| E | 4 | 4 |
| F | 3 | 7 |
| G | 7 | 7 |
| H | 7 | 11 |
| I | 12 | 12 |
| J | 13 | 13 |

En hier nog een keer onze zelf berekende uitslag:

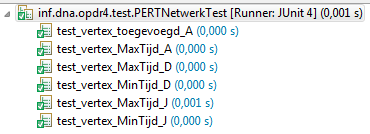
Het programma geeft het zelfde resultaat als onze eigen berekening zoals valt te zien in de uitvoer welke wij ook even in de hiernaast staande tabel hebben gezet.

Hiervoor is behalve het handmatig vergelijken van de gegevens ook een aantal JUnit testen gemaakt

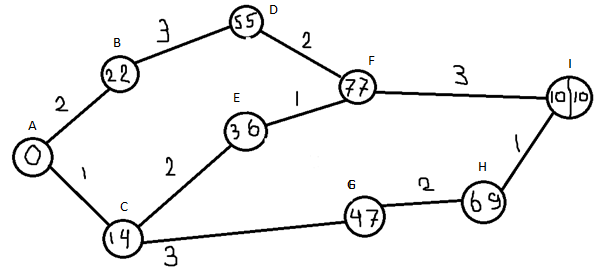


Bovenstaande testen testen de minimale en maximale tijd van punt D, de minimale tijd moet 2 zijn en de maximale tijd 5. In onderstaande afbeelding valt te zien dat dit klopt, de testen zijn geslaagd.

Ook valt er te zien dat de andere testen ook zijn geslaagd waardoor wij kunnen concluderen dat het goed wordt opgebouwd.

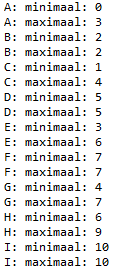


# Tweede voorbeeld:



In onderstaande tabel staat onze verwachte uitkomst met daaronder de daadwerkelijke uitvoer van de printfunctie in de applicatie

|  |  |  |
| --- | --- | --- |
| Knoop | Vroegste tijd | Laatste tijd |
| A | 0 | 0 |
| B | 2 | 2 |
| C | 1 | 4 |
| D | 5 | 5 |
| E | 3 | 6 |
| F | 7 | 7 |
| G | 4 | 7 |
| H | 6 | 9 |
| I | 10 | 10 |



Zoals valt te zien klopt de uitvoer met de verwachtingen en daaruit concluderen we dat ook deze test is geslaagd.