

SEMINARSKA NALOGA ZA PREDMET

UMETNA INTELIGENCA

Matic Šuc, 63180290

Domen Rupnik, 63180254

Fakulteta za računalništvo in informatiko, 2. letnik

Ljubljana, januar 2020

1. **Iskanje v globino**

Kot prvi implementirani algoritem, sva izbrala Iskanje v globino. Pot poišče tako, da najprej razvija najbolj globlje še nerazvito vozlišče. V najinem primeru je algoritem napisan tako, da se najprej premika gor, nato levo, potem desno in na koncu dol.

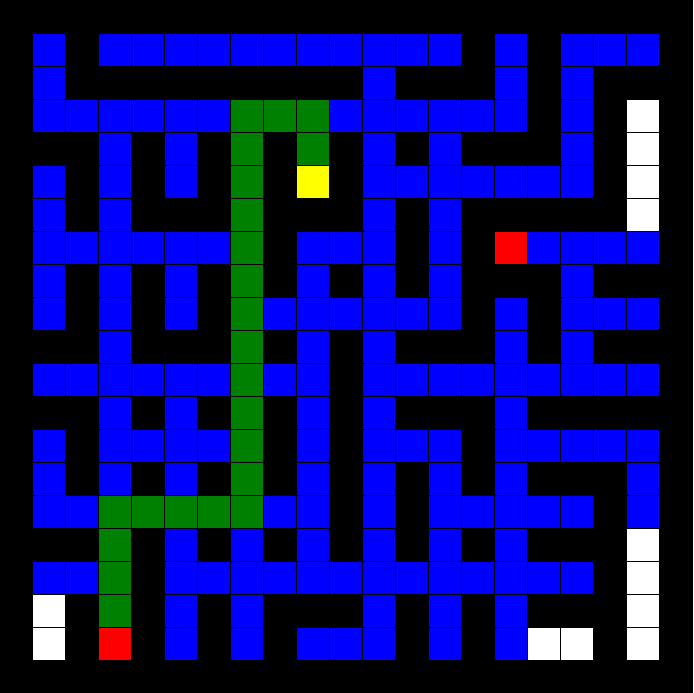


Rezultat za labirint št. 10 je:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število pregledanih vozlišč** |
| 103 | 53 | 144 |

1. **Iskanje v širino**

Kot drugi algoritem je bil izbran algoritem Iskanje v širino. Algoritem deluje tako, da se na vsakem koraku pregleda najbolj plitvo še ne razvito vozlišče.

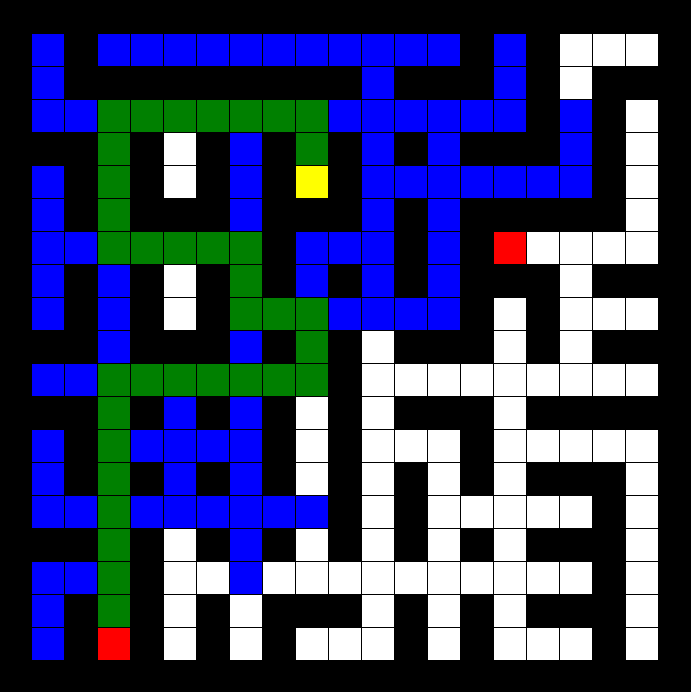


Rezultat za labirint št. 10 je:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število pregledanih vozlišč** |
| 55 | 25 | 205 |

1. **Iterativno poglabljanje**

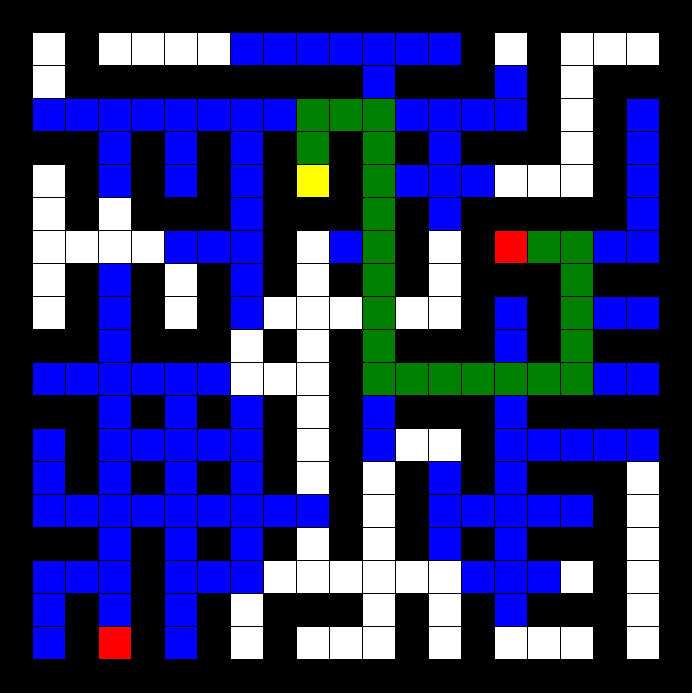
Iterativno poglabljanje deluje tako, da na vsakem koraku izvedemo iskanje v globino, z omejitvijo globine, ki jo na vsakem koraku povečamo.



|  |  |  |
| --- | --- | --- |
| **Cena pot** | **Dolžina poti** | **Število razvitih vozlišč** |
| 78 | 37 | 6590 |

1. **Dvosmerno iskanje**

Kot četrti algoritem je implementiran algoritem, ki deluje po principu iskanja v širino, ampak poteka iskanje z začetne točke in z vseh končnih točk.

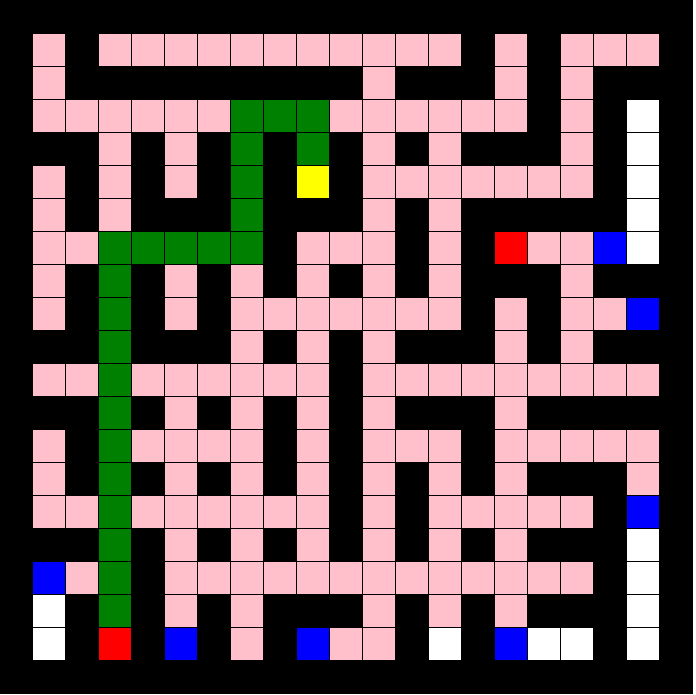


Rešitev za labirint št. 10:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število pregledanih vozlišč** |
| 55 | 25 | 144 |

1. **Dijkstra**

Algoritem Dijkstra deluje tako, da na vsakem koraku odpremo sosede vozlišča, na katerem smo trenutno ter trenutno vozlišče zapremo. Med odprtimi izberemo tistega, ki ima najmanjšo vrednost. Vrednosti se seštevajo glede na vrednosti vseh vozlišč, po katerih smo prišli do trenutnega vozlišča.

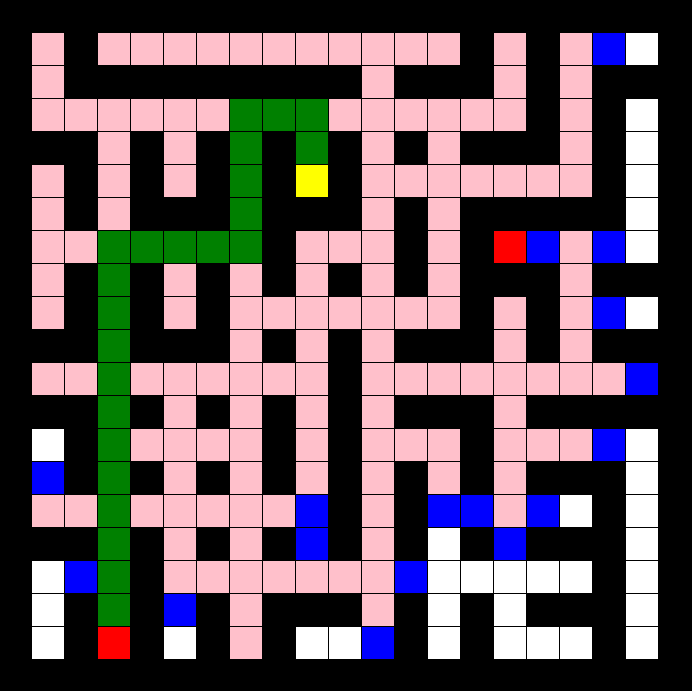


Rešitev za labirint št. 10:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število razvitih vozlišč** |
| 53 | 25 | 244 |

1. **A\***

Algoritem A\* deluje po istem principu kot Dijkstra, vendar pri izbiri vozlišča, ki bo naslednje zaprto, upošteva tudi hevristično oceno vozlišča. V tem primeru je hevristika ocenjena po Manhattanski razdalji od vozlišča, do najbližjega cilja.



Rezultati za labirint št. 10:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število razvitih vozlišč** |
| 53 | 25 | 181 |

1. **Požrešno preiskovanje**

Kot predzadnji algoritem je izbrano Požrešno preiskovanje, ki deluje tako, do na vsakem koraku razvije tisto sosednje vozlišče trenutnega vozlišča, ki ima najmanjšo ceno.

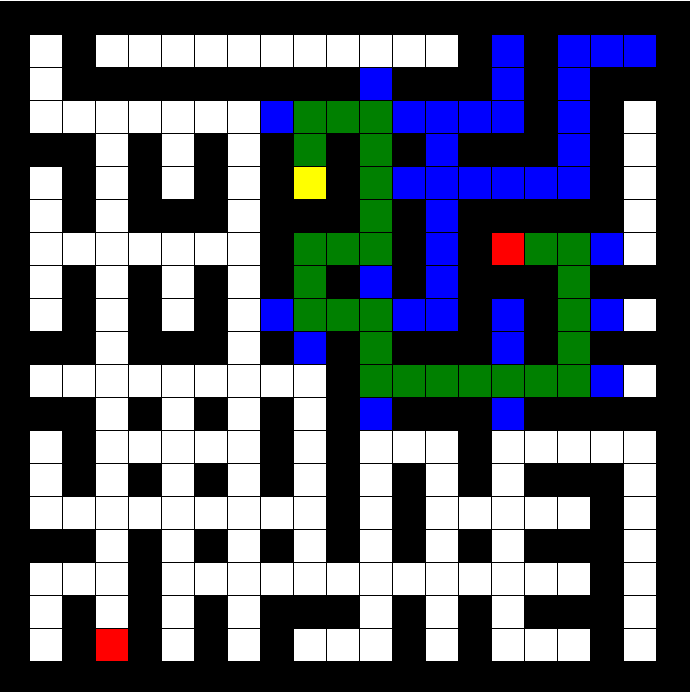


Rezultati za labirint št. 10:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število razvitih vozlišč** |
| 64 | 28 | 77 |

1. **Požrešno preiskovanje z uporabo hevristične ocene**

Požrešno preiskovanje nadgradimo tako, da pri izbiri najbolj optimalnega soseda, poleg cene do vozlišča štejemo še hevristično oceno. Hevristična ocena v tem primeru je Manhattanska razdalja do najbližjega končnega vozlišča.



Rezultati za labirint št. 10:

|  |  |  |
| --- | --- | --- |
| **Cena poti** | **Dolžina poti** | **Število razvitih vozlišč** |
| 64 | 28 | 65 |