به نام خدا

محمدصادق مجیدی یزدی شماره دانشجویی: ۹۸۱۰۶۰۰۴

فایل گزارش تمرین اول عملی استاد عبدی هجراندوست

توضیحات کلی

برای نمایش گرافیکی عملکرد الگوریتم‌های به کارگرفته‌شده در جستجوی باتری ربات، از منابع اینترنتی کار با رابط‌های گرافیکی زبان پایتون کمک گرفته شده است. محیط گرافیکی این برنامه با استفاده از کتابخانه شناخته‌شده pygame ایجاد شده و الگوریتم‌های نوشته‌شده با تعامل با این کتابخانه مراحل پیشرفت خود روی صفحه را مرحله به مرحله نمایش می‌دهند. در ابتدای اجرای برنامه با اجرای تابع parse_input_xml و پاس دادن آدرس فایل نمونه به آن در صورت لزوم(در صورتی که پاس ندهیم تابع با فرض اینکه نمونه در کنار فایل پایتون اصلی قرار گرفته است اجرا می‌شود)، با استفاده از ابزار کتابخانه lxml فایل را دی‌سریالایز کرده و سپس با استفاده از تابع make_grid اطلاعات خوانده شده را به یک آرایه دوبعدی از سلول‌ها (Cell) تبدیل می‌کنیم که فضای نمونه ما برای کار است. در ادامه با گرفتن کلیدهای مختلف از کاربر، الگوریتم‌های مختلف را برای جستجو روی صفحه اجرا می‌کند و یا صفحه را ریست می‌کند.

برای ایجاد تجربه منعطف‌تر برای استفاده از برنامه، دو متغیر سراسری در ابتدای کد به نام‌های WIDTH و DELAY_S تعریف شده است که به ترتیب عرض پنجره‌ی این برنامه و مقدار تاخیر بین اجرای هر گام الگوریتم‌ها بر حسب ثانیه تعیین می‌کنند. با تغییر هر کدام می‌توانید سایز پنجره را برای کار راحت‌تر تغییر داده یا سرعت نمایش تغییرات الگوریتم‌ها را متناسب با نظر خود تنظیم کنید.

فرض در نظر گرفته‌شده

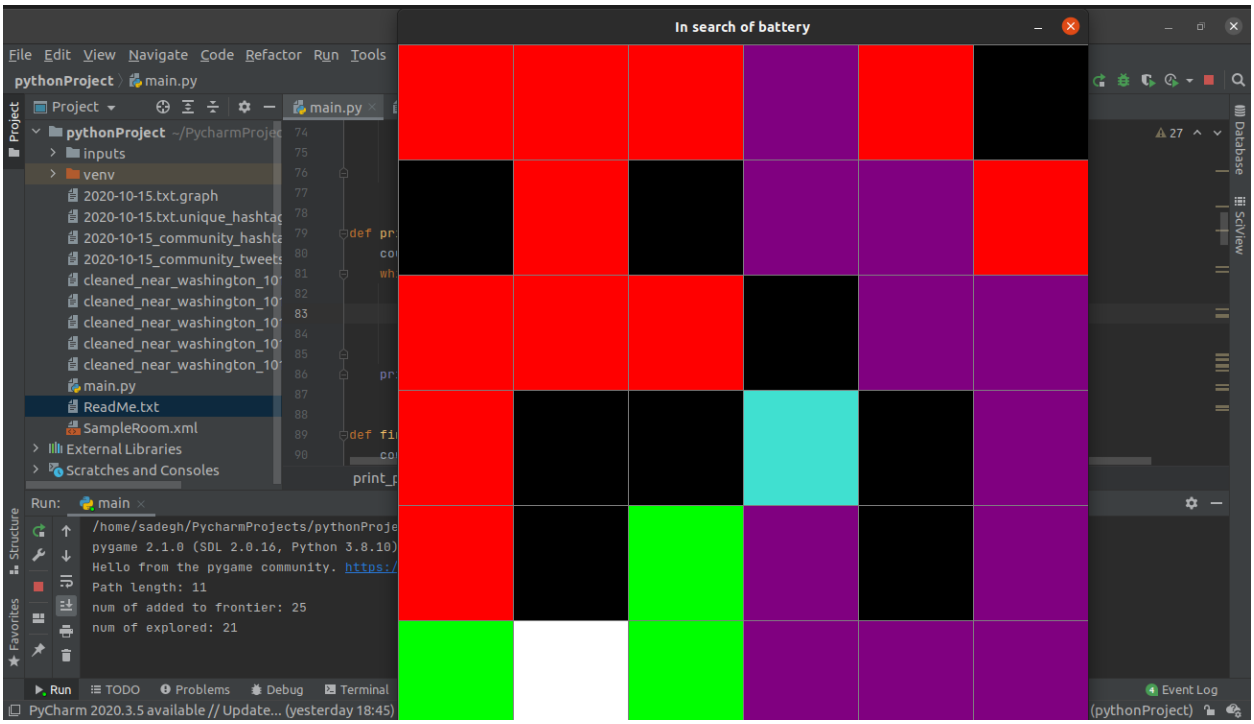
الگوریتم A* را با همان فرایض عادی، به صورتی که ربات در ابتدا فقط از محل شروع خود و از محل باتری صرفا برای تعیین مقدار تقریبی هیوریستیک آگاه است و از محیط اطراف خود دانش محدودی دارد. در هر مرحله با افزودن همسایه‌های بررسی نشده به مجموعه frontier خود دانش خود را از محیط توسعه داده و به ساخت مرحله به مرحله مسیر بهینه می‌پردازد. توجه داریم که ربات برای ساخت مسیر حرکتی انجام نمی‌دهد و صرفا به صورت انتزاعی مسیر بهینه را یافته و پس از اتمام به صورت مستقیم در آن مسیر حرکت می کند تا به باتری برسد. بنابراین هزینه اضافی رفت و برگشتی پرداخت نمی‌شود و جواب یافته شده هزینه بهینه را نشان می‌دهد.

بخش اول

در الگوریتم A* پیاده‌سازی‌شده از هیوریستیک منهتن استفاده شده است. قبلا در تمرین تئوری ثابت کردیم که منهتن، یک هیوریستیک قابل قبول است،. زیرا دقیقا مقدار کمینه حرکت‌های افقی و عمودی لازم برای رسیدن از مبدا به مقصد را نمایش می‌دهد. پس حتما مقدار هیوریستیک کمتر مساوی مقدار واقعی هزینه مسیر است. از طرفی چون حرکات ربات به صورت افقی یا عمودی است و پرش‌های ناگهانی و یا قطری نداریم و از طرفی هزینه هر جابجایی دقیقا برابر با ۱ است، بنابراین با رفتن به پسین یک خانه فعلی، مقدار هیوریستیک حداکثر یک واحد کم می‌شود. بنابراین مجموع هزینه مسیر و هیوریستیک پسین بزرگ‌تر مساوی هیوریستیک کنونی است و شرط سازگاری برقرار است. بنابراین منهتن برای این سوال هیوریستیک مناسبی است.

بخش دوم

نتیجه اجرای جستجوی A* روی فایل نمونه داده‌شده به صورت زیر است:

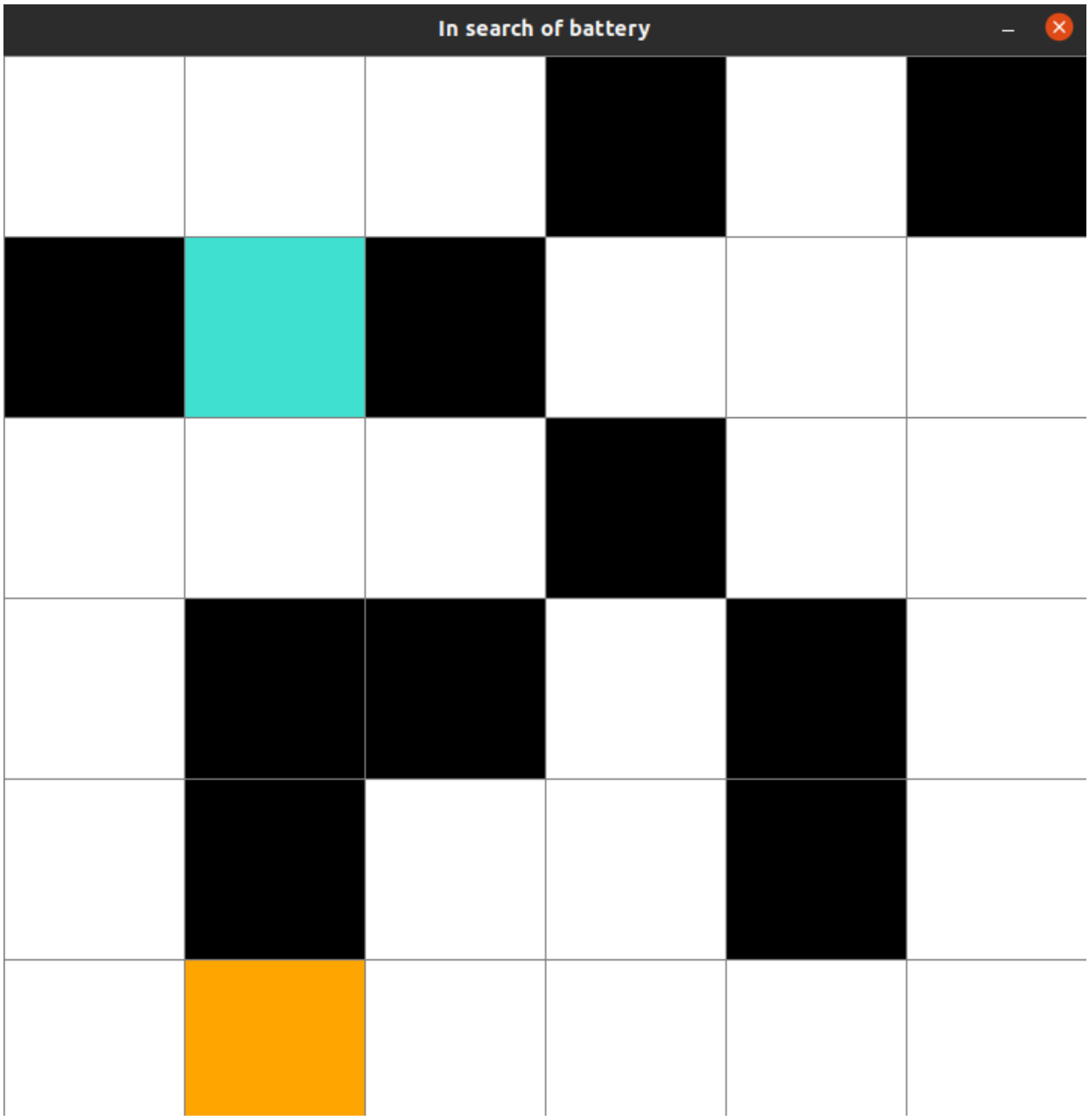


توجه کنید که در صفحه نمونه ما، رنگ سیاه نشان‌دهنده خانه‌های مانع، رنگ قرمز نشان دهنده خانه‌های بررسی‌شده یا explored، رنگ سبز نشان‌دهنده خانه‌های بررسی‌نشده مجموعه فرانتیر، رنگ سفید نشان‌دهنده خانه‌های بی‌تاثیر تا این مرحله جستجو و رنگ بنفش که پس از پایان جستجو روی صفحه ظاهر می‌شود نشان‌دهنده مسیر انتخابی ربات است.

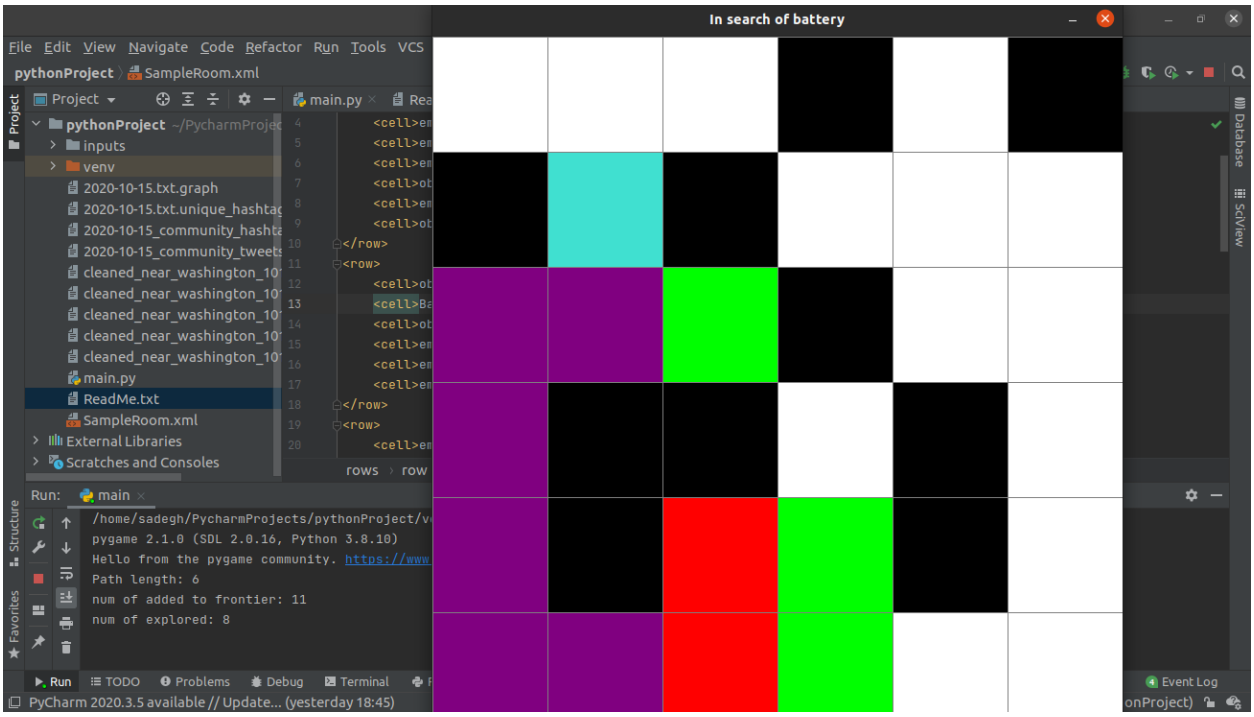
مشاهده می‌کنیم که روی فایل داده‌شده، تعداد ۲۵ خانه به مجموعه frontier اضافه شده و تعداد ۲۲ خانه explored شده و مسیر بهینه دارای طول ۱۱ است. با اجرای الگوریتم روی نمونه می‌توان تغییرات frontier و explored در هر مرحله را به صورت عینی بررسی کرد.

بخش سوم

صفحه اولیه جستجو را به صورت زیر تغییر می‌دهیم. دقت کنید رنگ نارنجی نشان‌دهنده مکان اولیه ربات و رنگ آبی نشان‌گر مکان باتری است. نمونه آن را در زیر می‌بینیم:



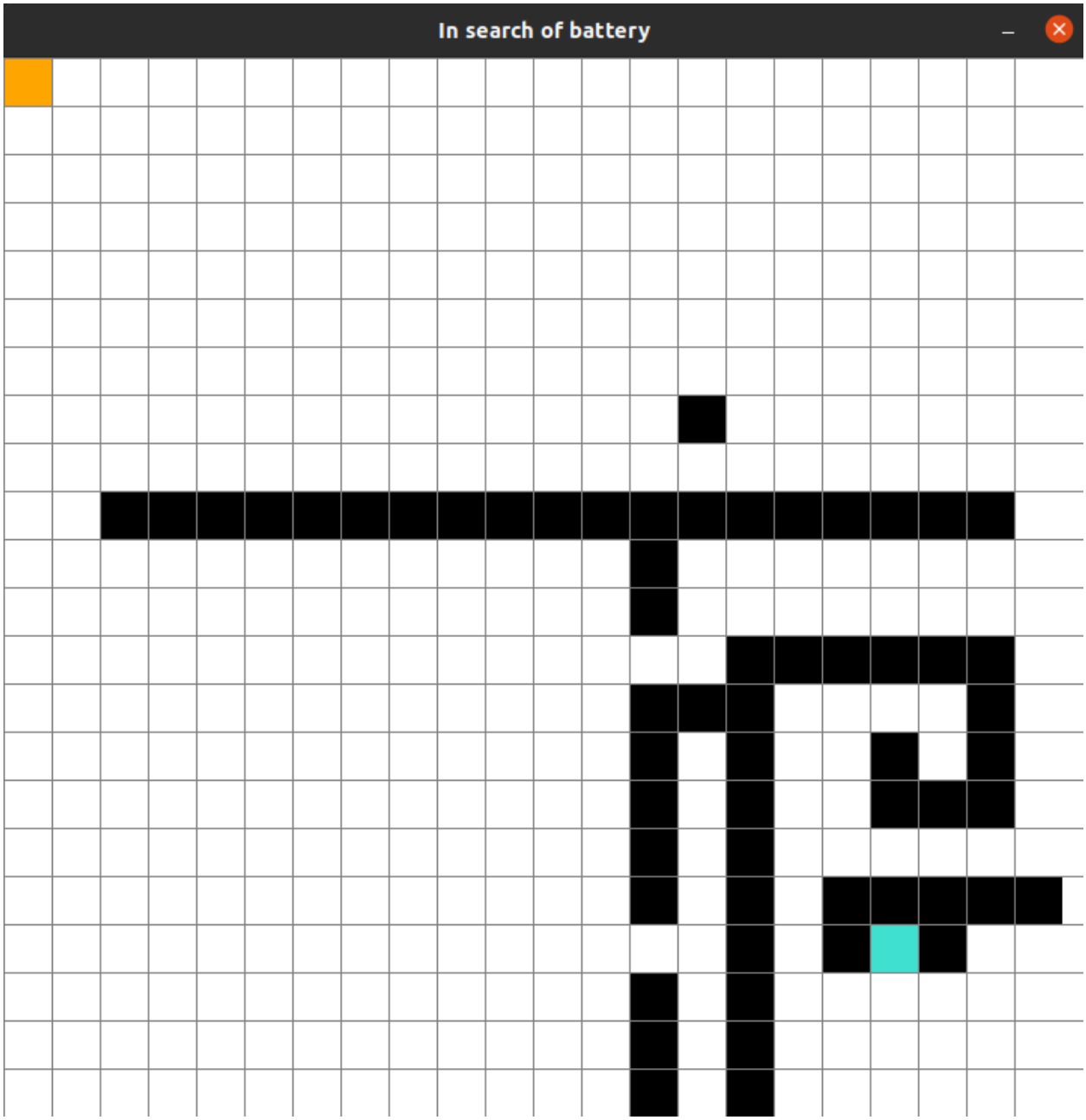
و همچنین نتیجه اجرای الگوریتم یافتن مسیر A* را روی آن در زیر مشاهده می کنیم:



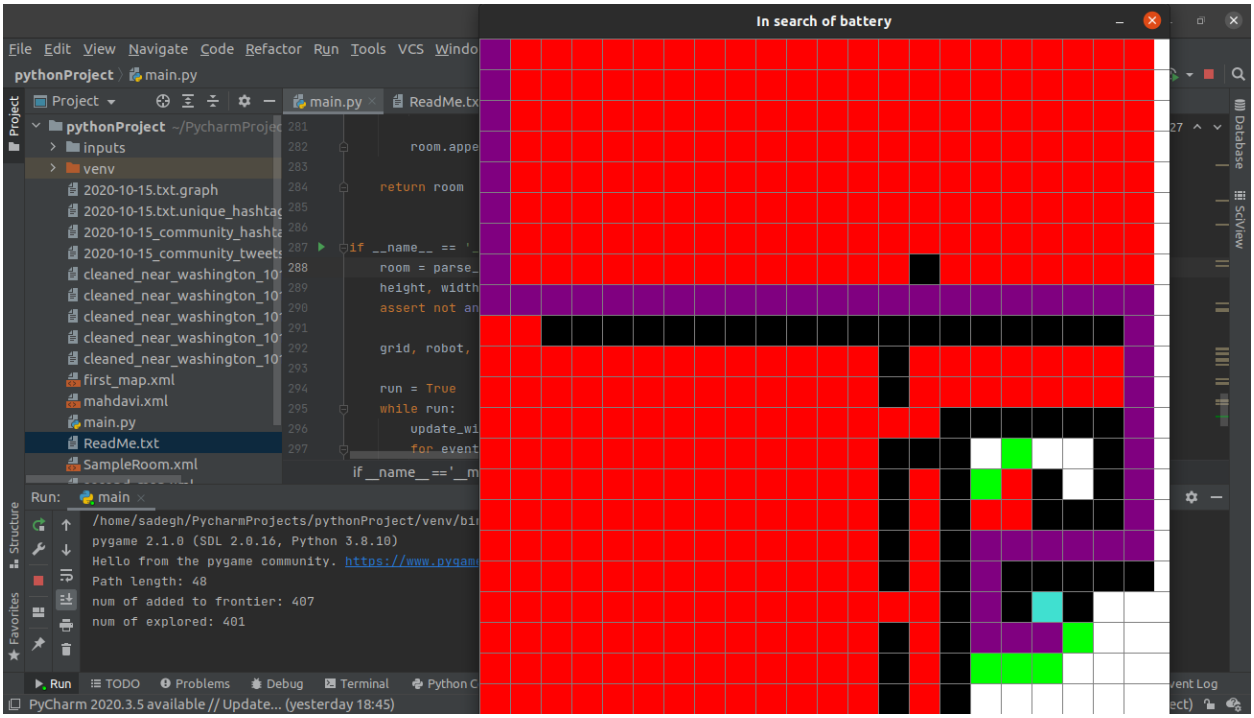
که مشخص است مسیر بهینه یافت شده و به لطف هیوریستیک مناسب تعداد کمی از خانه‌ها بررسی و بسط داده شده‌اند. در مجموع ۱۱ خانه به مجموعه frontier اضافه شده‌اند و تعداد ۸ خانه explored شده‌اند و مسیر بهینه به طول ۶ یافت می‌شود.

بخش چهارم

از چند نوع فایل ورودی دیگر برای نمونه استفاده می کنیم. (در این فایل‌ها از تست کیس‌های به اشتراک گذاشته‌شده دوستان دیگر بهره گرفته‌شده است)

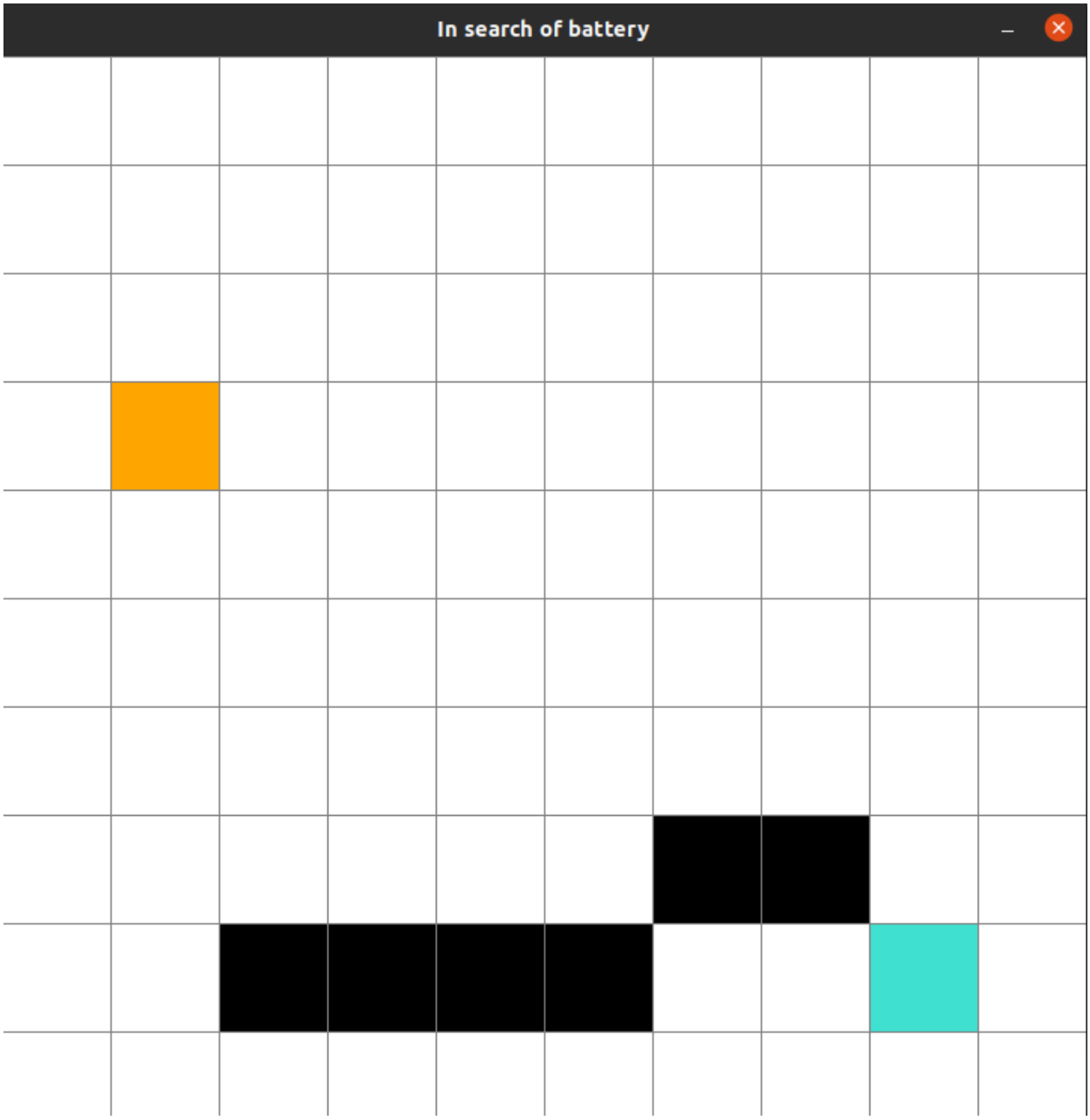


و نتیجه اجرای الگوریتم هیوریستیک منهتن روی آن به صورت زیر است:

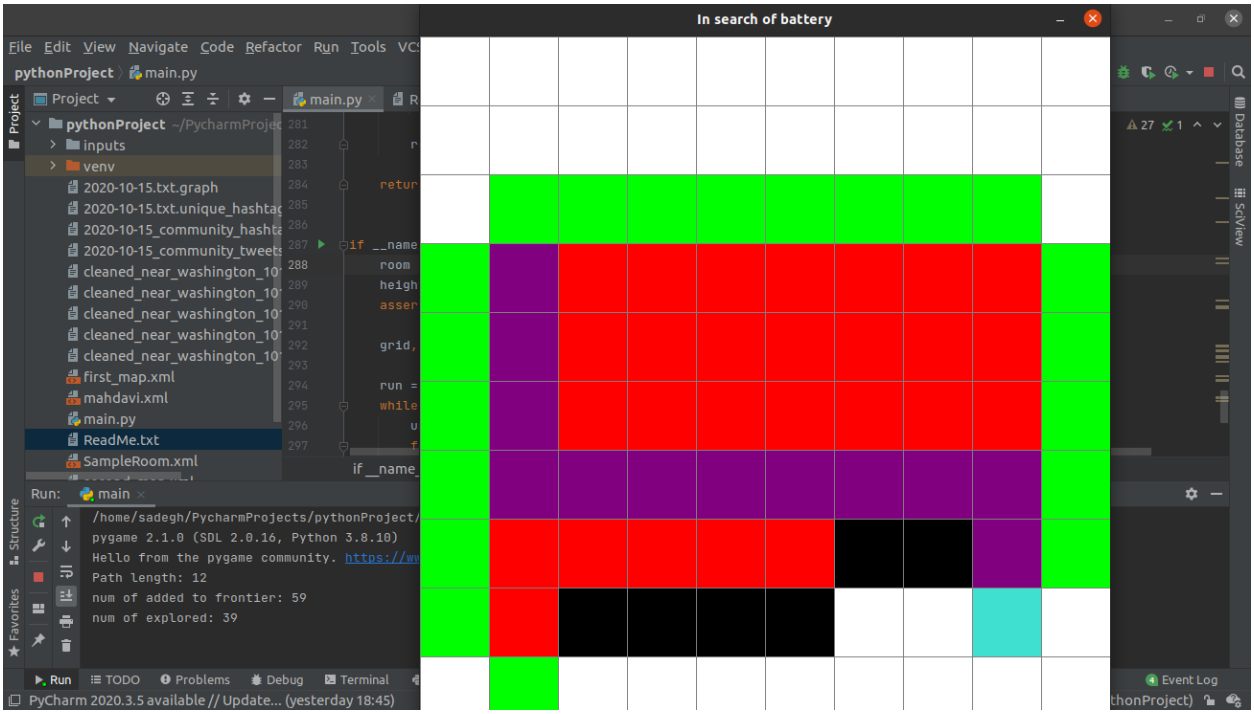


که مسیر بهینه به طول ۴۸ به صورت بالا روی آن مشخص شده است.

همچنین روی یک کیس دیگر نتیجه اجرای الگوریتم را بررسی می کنیم:



و نتیجه اجرای الگوریتم روی آن به صورت زیر است:



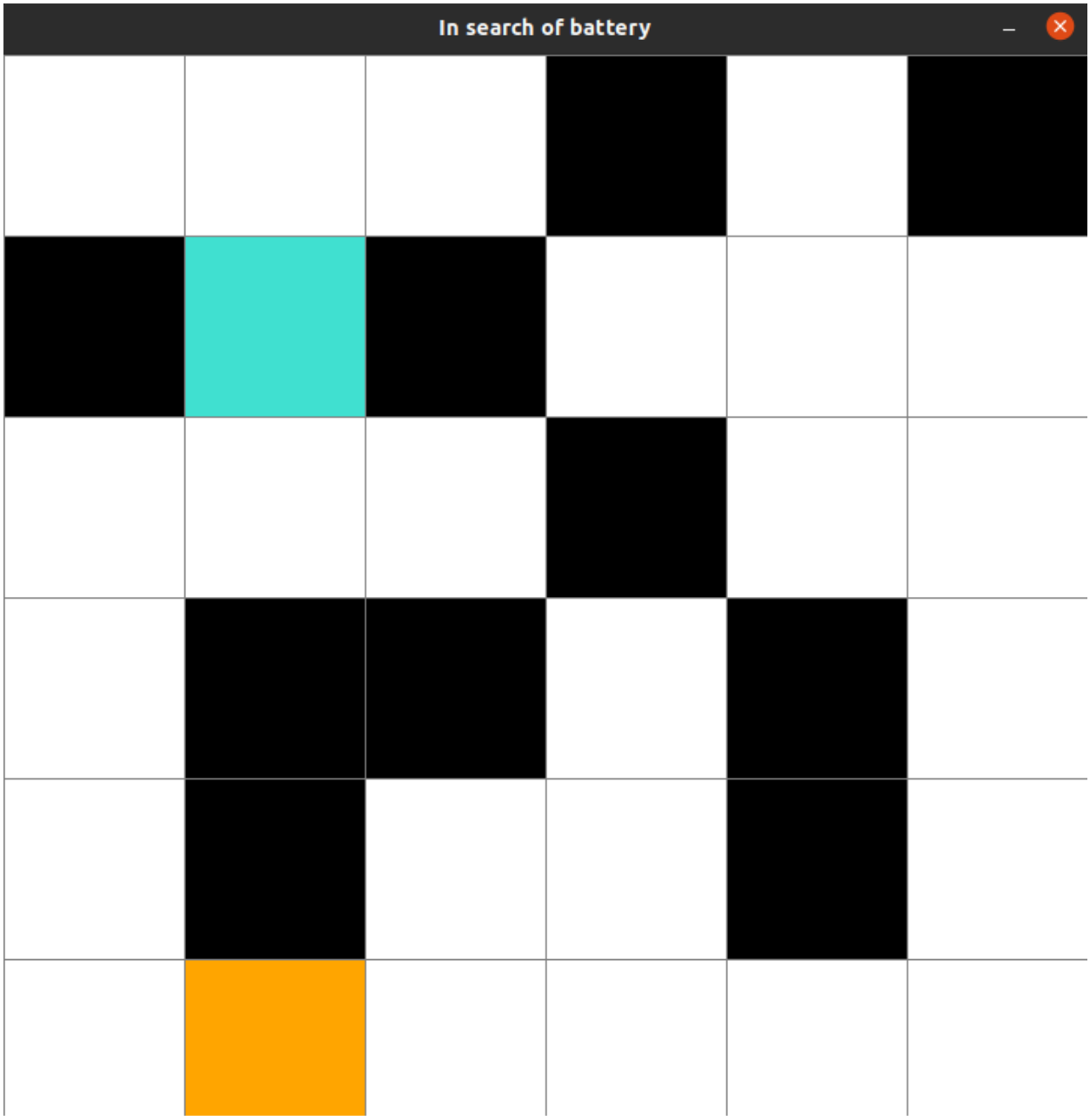
که مسیر بهینه به طول ۱۱ به صورت بالا پیدا شده است.

در صورتی که از یک هیوریستیک دیگر استفاده کنیم ممکن است در صورت غیر قابل قبول بودن مسیری نابهینه یافت شود و یا حتی مسیر بهینه با تعداد تلاش و بسط بیشتر پیدا شود. در این پیاده‌شازی به هیوریستیک دیگری نپرداخته‌ایم تا قادر به مقایسه آن با هیوریستیک منهتن باشیم.

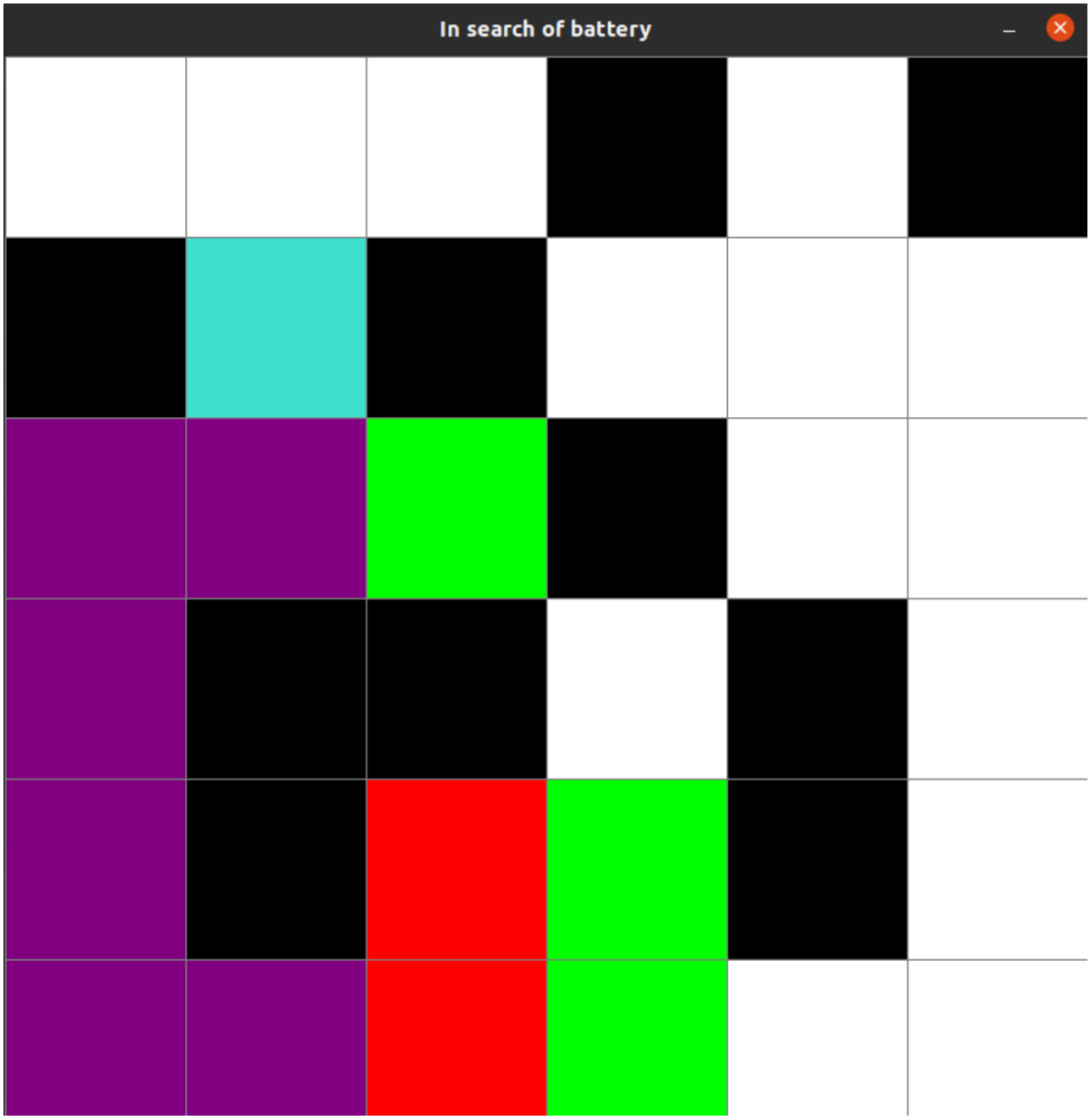
بخش پنجم

الگوریتم‌های جستجوی هزینه یکنواخت UCS و اول سطح BFS را پیاده‌سازی کرده‌ایم. آن‌ها را به همراه جستجوی اصلی روی یک صفحه مشترک انجام می‌دهیم و نتایج را مقایسه می کنیم.

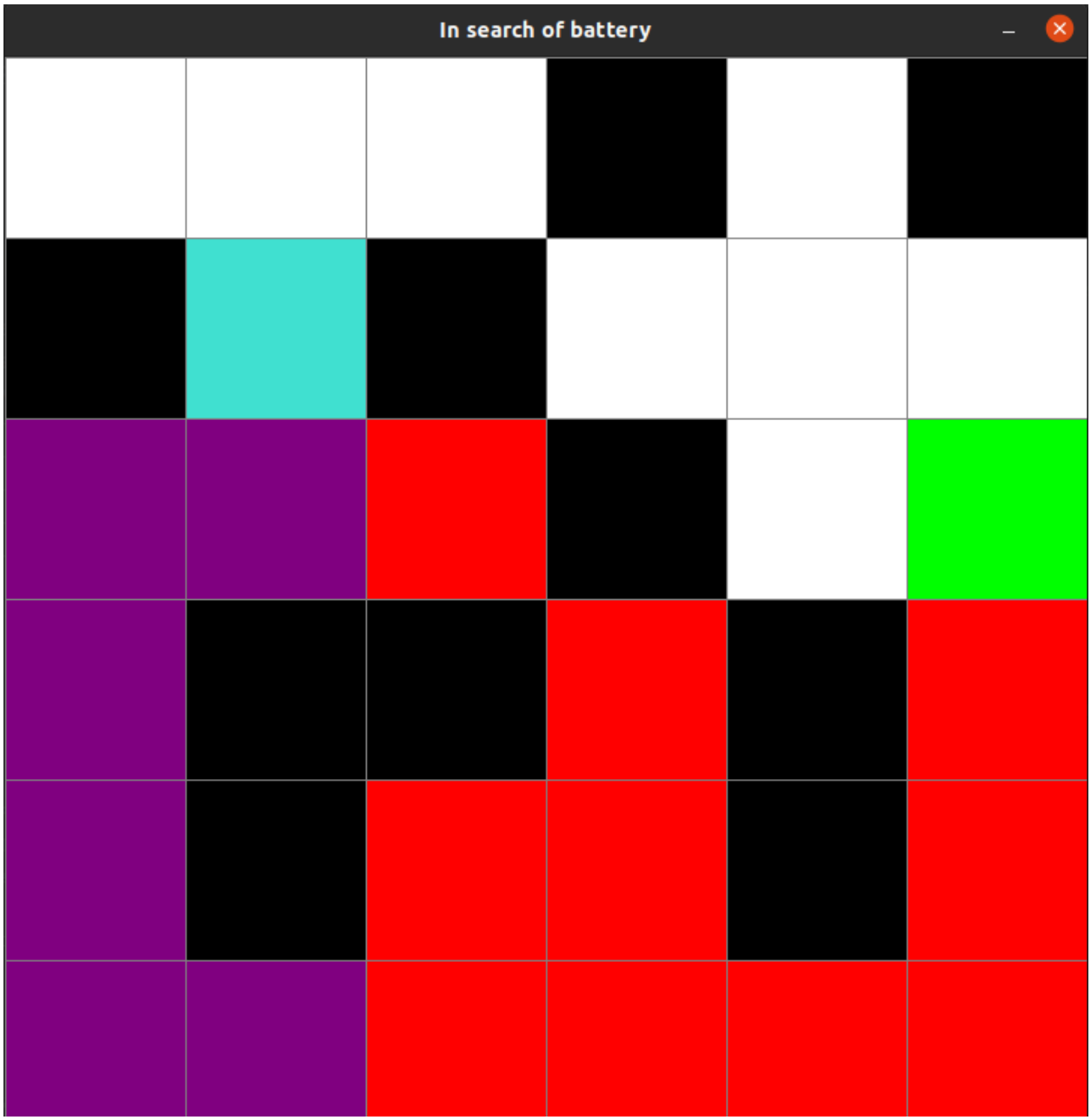
نقشه اولیه:



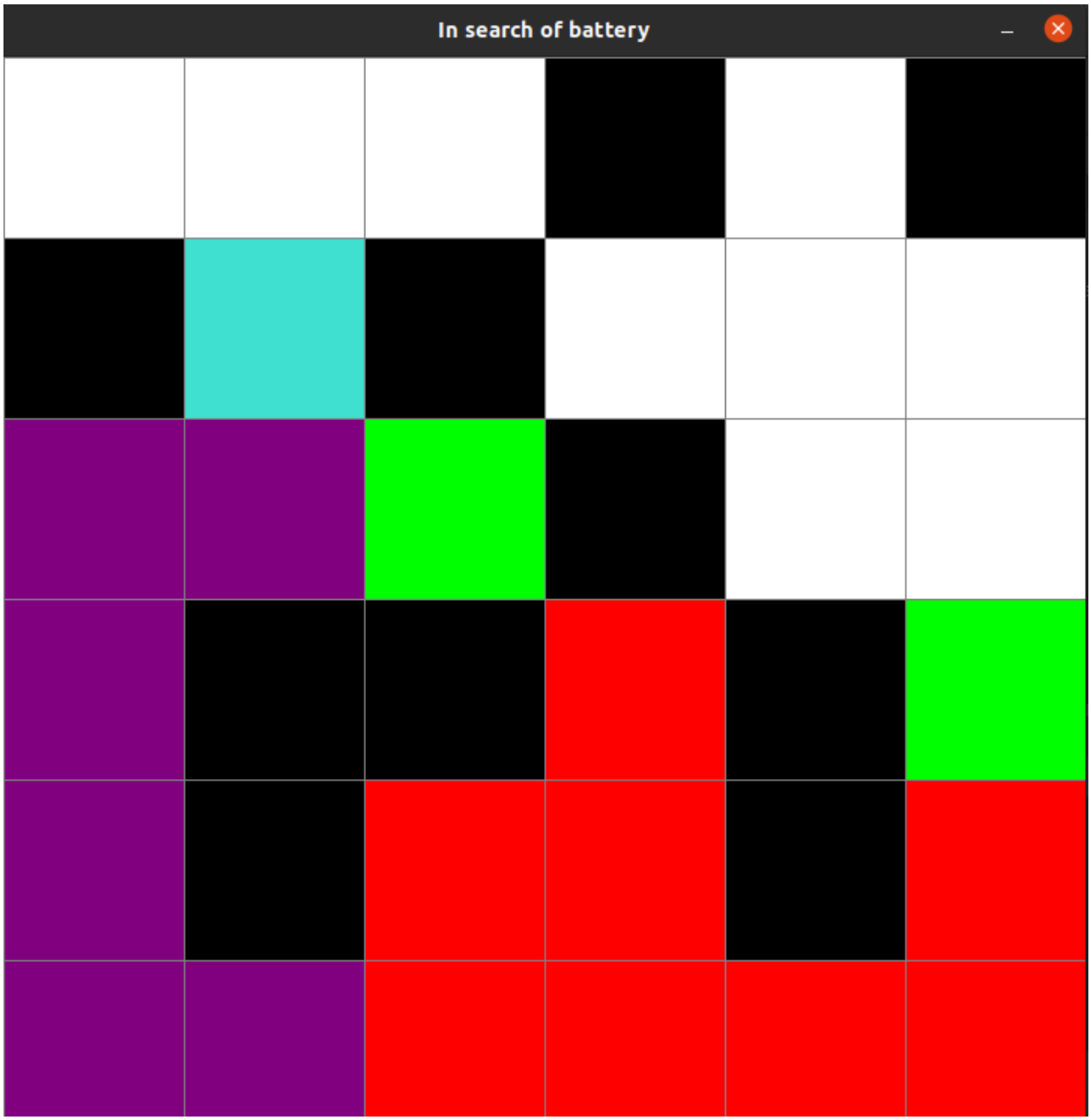
نتیجه اجرای A*:



نتیجه اجرای هزینه یکنواخت UCS:



و در نهایت نتیجه اجرای اول سطح BFS:



مشاهده می‌کنیم که هر دو حالت غیر از A* عملکرد ضعیف‌تر و مدت زمان اجرای بیشتری نسبت به A* داشته‌اند. البته با توجه به کامل و بهینه بودن هر ۳ همگی به یک مسیر بهینه رسیده‌اند ولی A* در کل به کمک هیوریستیک مناسب خود با بسط کمتر مسیر بهینه را یافته است.