2022-12-28
Autor: Robert Okuła
robert.okula@pg.edu.pl

1 Podstawy gry na Atari o nazwie Spy Hunter

Celem projektu jest napisanie aplikacji desktopowej będącą implementację gry na Atari o nazwie **Spy Hunter**. Celem gry jest jazda wzdłuż drogi i niszczenie innych (wrogich) pojazdów, wykorzystując broń która jest dostępna dla gracza. Projekt bazuje na wersji gry na Atari 2600. Więcej informacji na temat gry jest dostępne w Wikipedii. Przykład rozgrywki jest dostępny na Youtube.

2 Ogólne wskazania dotyczące projektu

Do zadania dołączony jest przykładowy projekt (szablon) w którym została zaimplementowana następująca funkcjonalność:

- obliczanie przyrostu czasu, co pozwala na śledzenie biegu programu;
- wyświetlanie obrazów w formacie BMP;
- rysowanie punktu (piksela), linii, prostokąta;
- wyświetlanie tekstu.

Szablon wykorzystuje bibliotekę „graficzną" SDL2 (2.0.7) – http://www.libsdl.org/. Jest ona załączona w projekcie rozruchowym i nie musi być uzyskana (pobierana) osobno.
Kompilację można wykonać wykorzystując polecenie (na 32-bitowym systemie operacyjnym Linux):
`g++ -O2 -I./SDL-2.0.7/include -L. -o main main.cpp -lm -lSDL2 -lpthread -ldl -lrt`
lub (na 64-bitowym systemie operacyjnym Linux):
`g++ -O2 -I./SDL-2.0.7/include -L. -o main main.cpp -lm -lSDL2x64 -lpthread -ldl -lrt`
W celu pomyślnego zbudowania szablonu, w tym samym folderze w którym znajduje się plik `main.cpp` powinny znajdować się dodatkowe zasoby:

- bitmapy (e.g. `cs8x8.bmp`, `eti.bmp`). Proszę zwrócić uwagę na rozmiar liter w nazwach plików!
- `libSDL2.a` (`libSDL2x64.a` na 64-bitowym systemie);
- folder `sdl`.

Projekt zawier skrypty, które mogą być wykorzystane do kompilacji (comp w środowisku 32-bitowym oraz comp64 w środowisku 64-bitowym).

Ocena projektu może zostać przeprowadzona w środowisku wybranym przez studenta. Do wyboru są dwie opcje:

- Linux – student jest zobowiązany do zweryfikowania, że program kompiluje się poprawnie i uruchamia się na dystrybucji zainstalowanej w laboratorium komputerowym przed rozpoczęciem oceniania;
- Windows – wykorzystując środowisko MS Visual C++ dostępne w laboratorium.

2.1 Obsługa programu

Program powinien wykorzystywać klawiaturę w następujący sposób:

- `strzałki`: poruszanie graczem na planszy;
- `esc`: zamknięcie programu;
- `n`: rozpoczęcie nowej gry;
- `s`: zapis stanu gry;

- `l`: wczytanie stanu gry;
- `p`: pauza/kontynuuj;
- `spacja`: strzelanie;
- `f`: zakończenie gry.

2.2 Wymagania obowiązkowe (5 punktów)

Wszystkie wymienione tutaj elementy należy zaimplementować. Brak któregokolwiek z poniższych elementów skutkuje otrzymaniem 0 pkt. z tego projektu.

(a) Plansza powinna być zaprezentowana w sposób estetyczny i ergonomiczny. Górna przestrzeń okna programu powinna zawierać imię, nazwisko oraz numer indeksu autora.

(b) Wyświetlanie upływajacego czasu i wyniku podczas rozgrywki. Obie wartości są resetowane przy rozpoczęciu nowej gry.

(c) Podstawowa funkcjonalność gry: ruch, droga (z kolizjami). Ruch jest natychmiastowy – reagowanie na wydarzenia.

(d) Obsługa klawiszy `esc` oraz `n` zgodnie z Rozdziałem 2.1.

(e) Gra powinna ponadto trzymać wynik gry zgodnie z zasadami mechaniki gry opisanymi w załączonym artykule.

(f) W prawym dolnym rogu ekranu gry powinny znajdować się litery odpowiadające zaimplementowanym elementom.

2.3 Wymagania nieobowiązkowe (11 punkty)

Poniższe elementy powinny być zaimplementowane zgodnie z zasadami opisanymi w artykule w Rozdziale 1 i w dodatkowych materiałach:

(g) (**2 pt.**) *Zapis i przywrócenie stanu gry.* Zapisz i załaduj stan gry z pliku. (klawisze `s` i `l`). Zapisana gra powinna być opisana przy użyciu czasu zapisu (daty, godziny, minuty i sekundy). Podczas ładowania lista zapisanych gier jest wyświetlana i użytkownik może wybrać pozycję w sposób wybrany przez autora.

(h) (**1 pt.**) *Wstrzymywanie gry.* Gracz powinien móc wstrzymać grę, a następnie powrócić do rozgrywki.

(i) (**2 pt.**) *Przeciwnicy i inne pojazdy.* Rozgrywka powinna zawierać przeciwników, których gracz może gonić, które mogą zaatakować i najeżdżać gracza, ale także zwykłe, niewrogie pojazdy, które w przypadku zniszczenia ich przez gracza zatrzymują wskaźnik punktacji gracza na chwilę.

(j) (**1 pt.**) *Strzelanie.* Gracz może strzelać zarówno do pojazdów wrogich i niewrogich, powinno to wpływać na wynik gracza w sposób opisany w załączonym artykule.

(k) (**1 pt.**) *Wypychanie z drogi.* Gracz może zmusić zarówno pojazdy wrogie jak i niewrogie z drogi, powinno to wpływać na wynik gracza w sposób opisany w załączonym artykule.

(l) (**1 pt.**) *Zdobywanie innych samochodów.* Przez krótki okres na początku rozgrywki gracz powinien mieć nielimitowaną liczbę samochodów (nawet jeżeli obecny samochód zostaje zniszczony, następny pojawia się natychmiastowo). Jednakże, po tym czasie liczba samochodów dostępna dla gracza powinna być ograniczona i nowe powinny być zdobywane w sposób opisany w artykule. Gdy liczba dostępnych aut wynosi zero, gra powinna się kończyć i ekran informujący gracza o tym fakcie powinien się wyświetlać.

(m) (**1 pt.**) *Power-up.* Na drodze może pojawić się inna broń (jest to pewne uproszczenie rozgrywki – nie musi występować Van z Bronią opisany w artykule) i gdy zostanie ona zebrana z jezdni (poprzez przejechanie po niej) tymczasowo (ograniczona ilość amunicji) zmienia sposób w jaki działa strzelanie (na przykład pozwala na strzelanie w dalszym zasięgu).

(n) (**2 pt.**) *Zachowywanie punktów.* Zachowuj i wyświetlaj listę najlepszych wyników. Jeden wynik składa się z liczby punktów i czasu gry. Gracz może edytować listę kończąc rozgrywkę. Lista jest ograniczona tylko przez rozmiar pamięci (dynamiczna alokacja pamięci). Lista najlepszych wyników powinna być stała (np. zapisywana w pliku i dostępna między uruchomieniami). Menu pozwala na wyświetlanie listy posortowanej po liczbie punktów (klawisz `p`) lub po czasie (klawisz `t`).

2.4 Wymagania dodatkowe (3 punkty)

Uwaga: poniższe wymagania są oceniane wyłącznie gry zaimplementowane zostały wszystkie poprzednie punkty (a)–(n) **na łączoną liczbę 16 punktów**.

(q) (**2 pt.**) *Gra dla dwóch graczy.* Autor powinien zaproponować formę kontroli drugiego gracza oraz formę rozgrywki dla dwóch graczy (kooperacja lub gracz przeciwko graczowi)?

(r) (**1 pt.**) *Inne power-upy.* Autor powinien zasugerować inne power-upy które mogą w pozytywny sposób wpłynąć na rozgrywkę.

2.5 Uwagi końcowe

- Używanie STLa jest zabronione.
- Kompilowanie i uruchamianie projektu jest wymagane do zdobycia jakichkolwiek punktów. Student musi sprawdzić czy komputer w laboratorium na którym będzie prezentować on swój projekt jest wyposażony w odpowiednie oprogramowanie.
- Projekt musi być zaimplementowany przez studenta samodzielnie. Student musi być w stanie wyjaśnić każdy element kody i być w stanie wprowadzić drobne zmiany na żądanie.
- Gra musi być grywalna, a program łatwy w obsłudze.