Polecenie	*Znaczenie*
1	Podnieś pióro
2	Opuść pióro
3	Obrót w prawo
4	Obrót w lewo
5,10	Przejdź 10 miejsc do przodu (lub liczbę inną niż 10)
6	Wyświetl tablicę 20 na 20
9	Koniec danych (znacznik)

Załóżmy, że żółw jest gdzieś w pobliżu środka podłogi. Poniższy „program" narysowałby i wyświetlił kwadrat 12 na 12 zostawiając pióro podniesione:

```
2
5,12
3
5,12
3
5,12
3
5,12
1
6
9
```

Kiedy żółw przemieszcza się z opuszczonym piórem, wstawiaj jedynki w odpowiednich elementach tablicy `podloga`. Kiedy zostanie wydane polecenie 6 (wyświetl), każdą 1 w tablicy zamień na gwiazdkę lub jakikolwiek inny wybrany znak. Gdziekolwiek jest zero, zostaw puste miejsce. Napisz program implementujący omówione tu możliwości grafiki. Utwórz kilka programów typu „grafika żółwia" rysujących interesujące kształty. Dodaj także inne polecenia, by zwiększyć możliwości swojej grafiki.

4.24 (*Przypadek Skoczka)* Jedną z bardziej interesujących łamigłówek dla szachowych zapaleńców jest przypadek Skoczka, pierwotnie zaproponowany przez znanego matematyka Eulera. Pytanie brzmi: Czy figura szachowa zwana Skoczkiem może przejść po pustej szachownicy dotykając każdego z 64 pól raz i tylko raz? Spróbujemy teraz głębiej omówić to intrygujące zadanie.

Skoczek wykonuje ruchy w kształcie litery L (dwa pola w jednym kierunku i jedno w kierunku poprzecznym). Tak więc z pola na środku pustej szachownicy skoczek może wykonać osiem różnych ruchów (ponumerowanych od 0 do 7) pokazanych na rysunku 4.25.

a) Narysuj na arkuszu papieru szachownicę 8 na 8 i spróbuj rozwiązać problem Skoczka ręcznie. Umieść 1 w pierwszym polu, na które się ruszasz, 2 na drugim, 3 na trzecim i tak dalej. Zanim zaczniesz przejście, oszacuj, jak daleko dojdziesz, pamiętając, że pełne przejście zawiera 64 ruchy. Jak daleko doszedłeś? Czy byłeś bliski swojemu przypuszczeniu?

	0	1	2	3	4	5	6	7
0								
1				2		1		
2			3				0	
3					K			
4			4				7	
5				5		6		
6								
7								

Rysunek 4.25 Osiem możliwych ruchów Skoczka

b) Teraz opracujmy program, który będzie poruszał Skoczkiem po szachownicy. Plansza jest reprezentowana przez dwuwymiarową tablicę 8 na 8 `plansza`. Każde pole jest zainicjowane wartością zero. Opisujemy każde z ośmiu możliwych ruchów za pomocą ich poziomych i pionowych składowych. Na przykład ruch typu 0 wg rysunku 4.25 zawiera przesunięcie o dwa pola poziomo w prawo i jedno pionowo w górę. Ruch 2 to przesunięcie o jedno pole poziomo w lewo i dwa pionowo w górę. Poziome ruchy w lewo i pionowe w górę są oznaczone liczbami ujemnymi. Osiem ruchów może być opisanych przez dwie jednowymiarowe tablice, `poziom` i `pion` tak jak przedstawiono niżej:

```
poziom[ 0 ] = 2
poziom[ 1 ] = 1
poziom[ 2 ] = -1
poziom[ 3 ] = -2
poziom[ 4 ] = -2
poziom[ 5 ] = -1
poziom[ 6 ] = 1
poziom[ 7 ] = 2

pion[ 0 ] = -1
pion[ 1 ] = -2
pion[ 2 ] = -2
pion[ 3 ] = -1
pion[ 4 ] = 1
pion[ 5 ] = 2
pion[ 6 ] = 2
pion[ 7 ] = 1
```

Niech zmienne `biezacyWiersz` i `biezacaKolumna` wskazują wiersz i kolumnę aktualnej pozycji skoczka. Chcąc wykonać ruch typu `numerRuchu`, gdzie `numerRuchu` mieści się w zakresie od 0 do 7, twój program używa instrukcji

```
biezacyWiersz += pion[ numerRuchu ];
biezacaKolumna += poziom[ numerRuchu ];
```

Wprowadź licznik zmieniający się od 1 do 64. Zapisz jego ostatni stan na każdym polu, na które przemieści się Skoczek. Pamiętaj o testowaniu każdego potencjalnego ruchu, by sprawdzić, czy bierka już odwiedziła to pole. I oczywiście obserwuj każdy potencjalny ruch, by upewnić się, że Skoczek nie wyląduje poza szachownicą. Teraz napisz program poruszający tą figurą po szachownicy. Uruchom go. Ile ruchów wykonał Skoczek?

c) Po próbie napisania i uruchomienia programu dotyczącego przypadku Skoczka, prawdopodobnie uzyskałeś pewną wartościową informację. Użyjemy jej do opracowania *heurystyki* (lub strategii) poruszania się Skoczka. Heurystyka nie gwarantuje sukces, ale starannie opracowana w dużym stopniu zwiększa szansę na sukcesu. Być może zauważyłeś, że zewnętrzne pola są bardziej kłopotliwe od tych w pobliżu środka planszy, a najbardziej niedostępnymi są cztery pola narożne.

Intuicja może podpowiadać, że powinieneś najpierw przemieścić skoczka na najbardziej kłopotliwe pola i zostawić otwarte te łatwiejsze, by blisko końca przejścia, kiedy plansza będzie coraz bardziej zatłoczona była większa szansa sukcesu.

Możemy opracować „heurystykę dostępnościową", klasyfikując każde z pól według tego, jak jest ono dostępne, a następnie przemieszczać Skoczka na najtrudniejsze pole (oczywiście zgodnie z jego ruchami w kształcie L). Opisujemy dwuwymiarową tablicę `dostepnosc` liczbami określającymi, z ilu pól wybrane pole jest dostępne. Na pustej szachownicy każde z centralnych pól jest klasyfikowane jako 8, każde z narożnych jako 2, a każde inne ma liczbę dostępności 3, 4 lub 6, jak poniżej:

```
2 3 4 4 4 4 3 2
3 4 6 6 6 6 4 3
4 6 8 8 8 8 6 4
4 6 8 8 8 8 6 4
4 6 8 8 8 8 6 4
4 6 8 8 8 8 6 4
3 4 6 6 6 6 4 3
2 3 4 4 4 4 3 2
```

Teraz napisz wersję programu o zadaniu Skoczka używając heurystyki dostępnościowej. Za każdym razem figura ta powinna przesunąć się na pole o najmniejszej liczbie dostępności. W razie równorzędności Skoczek może przemieścić się na dowolne z tych pól. Zatem przejście może zacząć się w dowolnym z czterech rogów. (*Uwaga:* Kiedy Skoczek będzie przemieszczał się po szachownicy, twój program powinien zmniejszać liczby dostępności w miarę zajmowania coraz większej liczby pól. W ten sposób w dowolnej chwili podczas przejścia liczba dostępności każdego pola będzie dokładnie równa liczbie pól, z których może być ono osiągnięte). Uruchom tę wersję programu. Otrzymałeś pełne przejście? Teraz zmodyfikuj program, aby wykonał 64 przejścia, po jednym zaczynającym się z każdego pola szachownicy. Ile pełnych przejść otrzymałeś?

d) Napisz wersję programu dotyczącego tego zadania, w której po napotkaniu równorzędności dwóch lub więcej pól, można zadecydować o wyborze pola, badając z wyprzedzeniem pola dostępne z „równorzędnych" pól. Twój program powinien znaleźć pole, z którego kolejny ruch prowadzi na następne z najmniejszą liczbą dostępności.

4.25 (*Przypadek Skoczka: metoda „brute force"*) W ćwiczeniu 4.24 opracowaliśmy rozwiązanie przypadku Skoczka. Zastosowane podejście, nazywane „heurystyką dostępności", tworzy wiele rozwiązań i działa efektywnie.

W miarę zwiększania się mocy obliczeniowej komputerów, będziemy mieli możliwość rozwiązywania większej ilości problemów z użyciem tylko mocy obliczeniowej i relatywnie nieskomplikowanych algorytmów. Tym razem spróbujmy rozwiązać ten problem tak zwaną metodą „brute force".

a) Użyj generatora liczb losowych, by pozwolić Skoczkowi na poruszanie się po szachownicy losowo (oczywiście jego prawidłowymi ruchami w kształcie L). Twój program powinien przeprowadzić jedno przejście i wyświetlić końcową szachownicę. Jak daleko doszedł Skoczek?

b) Najprawdopodobniej poprzedni program stworzył stosunkowo krótkie przejście. Teraz zmodyfikuj program tak, by przeprowadził 1000 przejść. Użyj tablicy jednowymiarowej do zapamiętywania liczby przejść każdej długości. Kiedy twój program zakończy 1000 przejść, powinien wyświetlić te informacje w prostym formacie tabelarycznym. Jaki był najlepszy wynik?

c) Najprawdopodobniej poprzedni program umożliwił kilka „poważnych" przejść, ale żadnego pełnego. Teraz nie hamuj się i po prostu pozwól mu pracować tak długo, dopóki nie wykona pełnego przejścia. (*Uwaga:* Ta wersja programu może pracować godzinami na potężnym komputerze). Użyj tablicy o liczbie przejść każdej długości i wyświetl ją, kiedy pierwsze pełne przejście zostanie znalezione. Ile przejść przeprowadził twój program przed wykonaniem pełnego przejścia? Ile czasu mu to zajęło?

d) Porównaj metodę „brute force" przypadku Skoczka z wersją heurystyki dostępnościowej. Która wymaga dokładniejszego przestudiowania problemu? Który algorytm jest trudniejszy do opracowania? Który wymaga większej mocy obliczeniowej? Czy możemy być wcześniej pewni, że otrzymamy pełne przejście za pomocą podejścia heurystyki dostępnościowej? Czy możemy być pewni (od razu), że otrzymamy pełne przejście za pomocą sposobu „brute force"? Przedyskutuj szczegółowo wszelkie wady i zalety rozwiązywania problemów tą metodą.

4.26 (*Ośmiu Hetmanów*) Inną łamigłówką dla szachowych zapaleńców jest przypadek Ośmiu Hetmanów. Krótko mówiąc: czy możliwe jest umieszczenie ośmiu Hetmanów na pustej szachownicy w ten sposób, by żaden z nich nie atakował innego, czyli żadna para tych figur nie może być w tym samym rzędzie, kolumnie ani przekątnej? Użyj sposobu opracowanego w ćwiczeniu 4.24, aby sformułować heurystykę do rozwiązania przypadku Ośmiu Hetmanów. Uruchom swój program. (*Wskazówka:* Jest możliwe przyporządkowanie do każdego pola szachownicy wartości wskazującej, ile pól pustej szachownicy zostanie „wyeliminowanych", jeśli Hetman zostanie postawiony na tym polu. Każdemu z pól narożnych byłaby przyporządkowana wartość 22 zgodnie z rysunkiem 4.26.) Kiedy tylko te „liczby eliminacji" zostaną umieszczone na wszystkich 64 polach, odpowiednią heurystyką będzie umieszczenie następnego hetmana na polu o najmniejszej liczbie eliminacji. Dlaczego ta strategia jest intuicyjnie ciekawa?