Zajęcia E-1: “Wprowadzenie do programowania dynamicznego”

Cel zajęć i efekty uczenia

Główne cele zajęć / materiał do opanowania:

- Programowanie dynamiczne: metoda bottom-up
- (opcjonalnie) Metoda spamiętywania (memoization)

Zadania do rozwiązania na sprawdzarce

Piramida
Plansza składa się z n kolejnych pól, numerowanych 1...n, z których niektóre są zablokowane (niedostępne). Pionek stoi na polu 1 i w każdym ruchu przemieszcza się do przodu co najmniej o jedno, a co najwyżej o 6 pól. Na ile sposobów może dojść do pola n, nie stając po drodze na żadnym zablokowanym polu?

Mosiężny Most
Dany jest ciąg n liczb. Wybrać z niego pewne liczby tak, żeby otrzymać jak największą sumę, nie wybierając nigdy trzech sąsiednich liczb.

Plan zajęć

Szacunkowy czas trwania: 2 godziny lekcyjne.

1. Metoda rekurencyjna i iteracyjna na obliczanie ciągu Fibonacciego
    - *Podstawowa wada metody rekurencyjnej: wielokrotne obliczanie tych samych wartości; metoda iteracyjna w bardzo oczywisty sposób oblicza każdą wartość tylko raz.*
2. Główna idea programowania dynamicznego - obliczyć podzadania jak w rekursji, ale tak, żeby żadnego nie obliczać wielokrotnie.
3. Metoda “top-down”: formułujemy podzadania - mniejsze wersje ostatecznego zadania - i ustawiamy je w łańcuch (w tym wypadku podzadania to wartości F(i)).
    - *Początkowe elementy łańcucha są łatwe do obliczenia (F(0) i F(1))*
    - *Każdy element da się obliczyć z poprzednich (F(i) z F(i-1) i F(i-2))*
    - *Ostatni element to nasze pierwotne zadanie (F(n)).*
    - *Metodę top-down polecam jako główny sposób przedstawiania algorytmów dynamicznych - jest dość intuicyjna i uniwersalna.*
4. [opcjonalnie] Metoda “bottom-up”, czyli spamiętywanie (nieoficjalnie ang. *memoization*):
    - *Wywołujemy normalną rekursję, ale przy wejściu do procedury sprawdzamy, czy wartość była już kiedyś obliczana.*

- *Jeśli nie, obliczamy ją rekurencyjnie i zapamiętujemy w tablicy (ew. STL-owej mapie).*
- *Jeśli już była i jej wartość jest w tablicy, zwracamy ją bez dalszych wywołań rekurencyjnych.*
- *Przykładowy pseudokod obliczania liczb Fibonacciego tą metodą:*

```
F[0..n] = {-1, -1, …, -1}

fibonacci(x):
        jeśli F[x] = -1:
                jeśli x = 0 to F[x] = 0
                jeśli x = 1 to F[x] = 1
                jeśli x > 1, to F[x] = fibonacci(x-1) + fibonacci(x-2)
                zwróć wynik F[x]
        jeśli F[x] ≠ -1:
                zwróć wynik F[x]
```

5. Zadanie “Piramida”.
   - *Jeden z najprostszych przykładów na programowanie dynamiczne: liczba sposobów S[k] dotarcia na pole k to 0, jeśli k jest zablokowane, i S[k-1]+...+S[k-6] w przeciwnym razie.*
6. Zadanie “Mosiężny most”
   - *Znowu, sformułowanie podzadania jest dość proste - jeśli tablica wejściowa to A[1..n], częściowy wynik W[k] to maksymalny zysk na początkowym fragmencie A[1..k].*
   - *Istotnym - i pouczającym - elementem zadania jest zauważenie, że wynik W[k] da się wyliczyć z poprzednich.*
   - *Jeśli nie wybieramy elementu A[k], to W[k] = W[k-1] - wynik jest taki sam, jak na elementach 1...k-1*
   - *Jeśli wybieramy A[k], ale nie wybieramy A[k-1], to W[k] = A[k] + W[k-2].*
   - *Jeśli wybieramy A[k] i A[k-1], to nie możemy wybrać A[k-2], więc wynik to W[k] = A[k]+A[k-1]+W[k-3].*