LMAD - Generowanie obiektów kombinatorycznych cz. 3 (zasada WW, tożsamości kombinatoryczne)

A Zadania na ćwiczenia

Zadanie A.1. Do sekcji koszykarskiej UAM zapisało się n osób, w tym m mężczyzn. Trener musi wybrać l osobową drużynę na turniej, przy czym zgodnie z regulaminem turnieju w składzie musi znaleźć się co najmniej jedna kobieta ($1 \leqslant l \leqslant m < n$). Zakładamy, że mamy do dyspozycji listę L osób zapisanych do sekcji, przy czym na tej liście pierwsze m osób to mężczyźni. Chcemy wygenerować wszystkie możliwe drużyny, które mogą być zgłoszone do turnieju. Poniżej przedstawiono dwa algorytmy, które rozwiązują ten problem:

1. Algorytm 1:
   (a) Generujemy wszystkie podzbiory l-elementowe zbioru n osób zapisanych do sekcji.
   (b) Generujemy wszystkie podzbiory l-elementowe zbioru m mężczyzn zapisanych do sekcji i usuwamy je z listy podzbiorów wygenerowanych w podpunkcie a).
2. Algorytm 2:
   (a) Wybieramy jedną liczbę k z wartości $\{m+1, m+2, \ldots, n\}$ i w skład drużyny włączamy kobietę, która znajduje się na liście osób zapisanych do sekcji na pozycji nr k.
   (b) Dla każdej wybranej liczby k generujemy wszystkie podzbiory $(l-1)$-elementowe zbioru osób zapisanych do sekcji, które na liście L znajdują się na pozycjach o numerach od 1 do $k-1$. Do każdego takiego podzbioru dorzucamy kobietę, która na liście L miała numer k.

Zaimplementuj oba te algorytmy dla $n = 6$, $m = 4$ i $l = 3$. Porównaj drużyny wygenerowane w obu przypadkach - czy zostały wygenerowane te same drużyny? Oblicz, ile drużyn wygenerowały oba algorytmy. Zastanów się, jak w przypadku dowolnych wartości n, m, l można by zapisać liczbę obiektów generowanych przez oba te algorytmy. Jaką tożsamość kombinatoryczną możemy udowodnić w ten sposób?

Zadanie A.2. Danych jest n różnych kul ponumerowanych liczbami od 1 do n. Chcemy pokolorować k z nich ($1 \leqslant k \leqslant n$), przy czym mamy do dyspozycji dwa kolory - niebieski i czerwony. Naszym celem jest wygenerowanie wszystkich możliwych pokolorowań. Pokolorowania będziemy zapisywać w formie list $[N, C, B]$, gdzie N oznacza listę kul pomalowanych na niebiesko, C listę kul pomalowanych na czerwono, a B listę kul niepomalowanych żadnym kolorem (wszystkie listy powinny być posortowane). Poniżej przedstawiono dwa algorytmy, które rozwiązują ten problem:

1. Algorytm 1:
   (a) Znajdujemy wszystkie k-elementowe podzbiory P zbioru $\{1, 2, \ldots, n\}$ odpowiadające kulom, które zostaną pomalowane.
   (b) Dla każdego wygenerowanego podzbioru P generujemy wszystkie jego uporządkowane podziały na dwa zbiory przechowywane w listach N i C zawierające kule pomalowane odpowiednio na niebiesko i czerwono. Do każdego takiego podziału $[N, C]$ dodajemy listę B zawierającą kule niepomalowane w tym przypadku (czyli te, które nie należą do zbioru P) otrzymując listę list $[N, C, B]$.
2. Algorytm 2:
   (a) Ustalamy liczbę kul pomalowanych na niebiesko poprzez wybranie liczby j ze zbioru $\{0, 1, \ldots, k\}$
   (b) Dla każdej liczby j generujemy wszystkie j-elementowe podzbiory zbioru kul, które umieszczamy na liście N. Będą to kule pomalowane na niebiesko.
   (c) Dla każdego N generujemy wszystkie $(k-j)$-elementowe podzbiory zbioru pozostałych kul i umieszczamy je na liście C. Będą to kule pomalowane na czerwono.
   (d) Dla każdej pary N i C wyznaczamy listę B kul niepomalowanych i ostatecznie otrzymujemy listę list $[N, C, B]$ zawierającą jedno z możliwych pokolorowań.

Zaimplementuj oba te algorytmy dla $n = 5$ i $k = 3$. Porównaj listy wygenerowane w obu przypadkach - czy zostały wygenerowane te same pokolorownia? Oblicz, ile list wygenerowały oba algorytmy. Zastanów się, jak w przypadku dowolnych wartości n i k można by zapisać liczbę obiektów generowanych przez oba te algorytmy. Jaką tożsamość kombinatoryczną możemy udowodnić w ten sposób?

Zadanie A.3. Danych jest n ($n \geqslant 3$) kul ponumerowanych kolejnymi liczbami naturalnymi: $1, 2, \ldots, n$, które chcemy umieścić w 3 rozróżnialnych urnach w taki sposób, aby każda urna zawierała co najmniej jedną kulę. Innymi słowy, chcemy znaleźć uporządkowane podziały zbioru kul na trzy niepuste podzbiory U_1, U_2, U_3 odpowiadające podzbiorom kul w poszczególnych urnach. Poniżej przedstawiono propozycje różnych algorytmów, które generują wszystkie takie możliwe rozmieszczenia kul:

1. Permutujemy kule o numerach od 1 do 3 otrzymując permutacje $P = [k_1, k_2, k_3]$. Następnie generujemy wszystkie możliwe rozmieszczenia pozostałych kul do urn, czyli listy podzbiorów $[U_1, U_2, U_3]$. Na koniec łączymy te rozmieszczeniami z permutacjami trzech pierwszych kul (tzn. do U_1 dodajemy kulę k_1, do U_2 dodajemy kulę k_2, a do U_3 dodajemy kulę k_3), aby dostać ostateczne konfiguracje.

2. Wybieramy podzbiór trzyelementowy zbioru kul i permutujemy kule z tego podzbioru otrzymując permutacje $P = [k_1, k_2, k_3]$. Następnie generujemy wszystkie możliwe rozmieszczenia pozostałych kul do urn, czyli listy podzbiorów $[U_1, U_2, U_3]$. Na koniec łączymy te rozmieszczeniami z permutacjami trzech pierwszych kul (tzn. do U_1 dodajemy kulę k_1, do U_2 dodajemy kulę k_2, a do U_3 dodajemy kulę k_3), aby dostać ostateczne konfiguracje.

3. Generujemy wszystkie całkowitoliczbowe rozwiązania równania:

$$x_1 + x_2 + x_3 = n,$$

   w których $x_i \geqslant 1$ dla każdego $i = 1, 2, 3$. Następnie dla każdego otrzymanego rozwiązania $[x_1, x_2, x_3]$ generujemy uporządkowany podział zbioru kul na trzy podzbiory U_1, U_2, U_3 o mocach równych odpowiednio x_1, x_2, x_3.

4. Generujemy wszystkie uporządkowane podziały zbioru kul na trzy podzbiory odpowiadające rozmieszczeniom kul w trzech urnach, a następnie jeśli zostały wygenerowane podziały zawierające zbiory puste, to wyrzucamy je.

Zaimplementuj każdy z powyższych algorytmów, a następnie dla każdego z nich:

a) sprawdź, jakie rozmieszczenia zostały wygenerowane - czy każde rozmieszczenie zostało wygenerowane i czy żadne rozmieszczenia się nie powtarzają?

b) oblicz, ile rozmieszczeń otrzymaliśmy,

c) spróbuj określić, jak wyglądałby wzór na liczbę rozmieszczeń generowanych tym algorytmem w ogólnym przypadku n kul i k urn ($n \geqslant k$),

d) określ, czy rozwiąnie problemu jest poprawne, a jeśli nie, to spróbuj określić dlaczego.

B Zadania na ćwiczenia - jeśli czas pozwoli

Zadanie B.1. Napisz algorytm, który wyznacza wszystkie najkrótsze drogi pomiędzy przeciwległymi wierzchołkami kraty o wymiarach $m \times n$, gdzie $n \leqslant m$. W tym celu możemy przyjąć, że wyznaczamy drogi pomiędzy lewym dolnym wierzchołkiem kraty $A = (0, 0)$, prowadzące do prawego górnego wierzchołka kraty $B = (m, n)$. Każdą taką drogę możemy reprezentować za pomocą sekwencji ruchów G i P oznaczających odpowiednio pójście w górę lub w prawo. Rozważmy dwa algorytmy, rozwiązujące ten problem:

1. Algorytm 1: Generujemy wszystkie ciągi długości $2n$ składające się z dokładnie n liter G i n liter P.

2. Algorytm 2:

   (a) Ustalamy liczbę k ze zbioru $\{0, 1, \ldots, n\}$.

   (b) Generujemy wszystkie drogi, które przechodzą przez punkt o współrzędnych $(k, n-k)$ (np. korzystając z pierwszego algorytmu). Proszę zwrócić uwagę, że dla $k \neq \ell$, najkrótsza droga z A do B przechodząca przez punkt $(k, n-k)$ nie może jednocześnie przechodzić przez punkt $(\ell, n-\ell)$ (dlaczego?).

Zaimplementuj oba te algorytmy dla $n = 4$ i $m = 5$. Porównaj drogi wygenerowane w obu przypadkach - czy zostały wygenerowane te same sekwencje? Oblicz, ile dróg wygenerowały oba algorytmy. Zastanów się, jak w przypadku dowolnych wartości n i m można zapisać liczbę obiektów generowanych przez obydwa algorytmy. Jaką tożsamość kombinatoryczną możemy udowodnić w ten sposób?

C Zadania do samodzielnej pracy w domu

Zadanie C.1. Student rozwiązuje test składający się z n pytań typu „Prawda/Fałsz". Student na karcie odpowiedzi przy każdym pytaniu może zaznaczyć jedną z liter „P" (jeśli uważa, że stwierdzenie jest prawdziwe) lub „F" (jeśli uważa, że stwierdzenie jest fałszywe) albo może nie zaznaczyć żadnej odpowiedzi, jeśli nie jest pewny, która z nich jest prawdziwa. Chcemy wygenerować wszystkie możliwe zestawy odpowiedzi studenta. W tym celu rozważymy dwa algorytmy rowiązujące ten problem:

1. Algorytm 1: Generujemy wszystkie ciągi długości n składające się z liter „P" (prawda), „F" (fałsz) lub „B" (brak odpowiedzi).

2. Algorytm 2:

(a) Ustalamy liczbę k ze zbioru $\{0, 1, \ldots, n\}$

(b) Dla każdej liczby k generujemy wszystkie k-elementowe podzbiory O zbioru pytań, odpowiadające pytaniom, na które student zdecyduje się udzielić odpowiedzi.

(c) Dla każdego podzbioru O generujemy wszystkie ciągi długości n, które na pozycjach o indeksach ze zbioru O będą miały litery „P" lub „F", a na pozostałych miejscach znajdą się litery „B".

Zaimplementuj oba te algorytmy dla $n = 5$. Porównaj listy wygenerowane w obu przypadkach - czy zostały wygenerowane te same ciągi odpowiedzi? Oblicz, ile ciągów odpowiedzi wygenerowały oba algorytmy. Zastanów się, jak w przypadku dowolnych wartości n można by zapisać liczbę obiektów generowanych przez oba te algorytmy. Jaką tożsamość kombinatoryczną możemy udowodnić w ten sposób?

Zadanie C.2. Dana jest grupa składająca się z n mężczyzn i m kobiet. Każda osoba z tej grupy kupuje jeden los na loterii. Chcemy wygenerować wszystkie możliwe podzbiory osób, które kupiły los zwycięski (zakładamy przy tym, że wygrać mogła dowolna liczba osób). W tym celu możemy zastosować dwa algorytmy:

1. Algorytm 1:

   (a) Ustalamy liczbę zwycięzców k ze zbioru $\{0, 1, \ldots, n + m\}$.

   (b) Dla każdej liczby k generujemy wszystkie podzbiory k-elementowe osób z tej grupy.

2. Algorytm 2:

   (a) Ustalamy liczbę k mężczyzn ze zwycięskim losem, gdzie $k \in \{0, 1, \ldots, n\}$

   (b) Dla każdego k generujemy wszystkie k-elementowe podzbiory M zbioru n mężczyzn z tej grupy.

   (c) Ustalamy liczbę j kobiet ze zwycięskim losem, gdzie $j \in \{0, 1, \ldots, m\}$.

   (d) Dla każdego j generujemy wszystkie j-elementowe podzbiory K zbioru m kobiet z tej grupy.

   (e) Dla każdego zbioru M i K wygenerowanych w poprzednich podpunktach generujemy nowy zbiór $Z = M \cup K$ zwycięzców.

Zaimplementuj oba te algorytmy dla $n = 3$ i $k = 4$. Porównaj listy wygenerowane w obu przypadkach - czy zostały wygenerowane te same zbiory zwycięzców? Oblicz, ile zbiorów zwycięzców wygenerowały oba algorytmy. Zastanów się, jak w przypadku dowolnych wartości n i k można by zapisać liczbę obiektów generowanych przez oba te algorytmy. Jaką tożsamość kombinatoryczną możemy udowodnić w ten sposób?

Zadanie C.3. Napisz program, który będzie generować wszystkie n-elementowe permutacje bez punktów stałych (nieporządki), a także wyznaczać ich liczbę. Sprawdź jego działanie dla $n = 6$.