Aufgabe 3: Eisbudendilemma

Teilnahme-ID: 59116

Bearbeiter/-in dieser Aufgabe:
Jim Maar

18. April 2021

Inhaltsverzeichnis

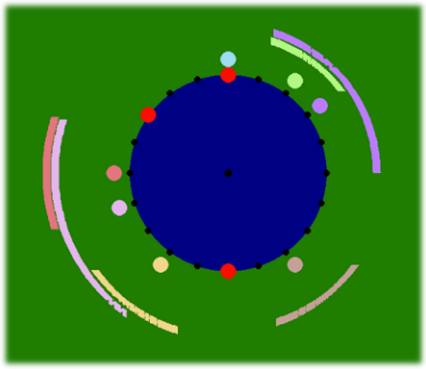
Lösungsidee

Das Problem wurde für eine beliebige Anzahl an Eisbuden über 1 und eine beliebige Anzahl an Häusern über 1 gelöst.

Zur Vereinfachung werden wir die Anzahl der Häuser H und die Anzahl der Eisbuden E nennen.

Jedes Haus hat einen Bereich, der von dem Haus bis einen Schritt vor die nächste Eisbude und dieselbe Strecke in die andere Richtung geht. Wenn eine Eisbude in diesen gelegt wird, stimmt der Bewohner des Hauses für die neuen Eisbudenstandorte. Der Bereich geht nicht ganz bis zur Eisbude, da die Eisbuden nur auf ganzzahligen Positionen stehen sollen (So habe ich es zumindest verstanden [39.2 Aufgabe3] Eisstände und Abstimmungen - Community (einstieg-informatik.de)).

Die Bereiche jedes Hauses werden im unteren Bild anhand eines Beispiels gezeigt.



Ob bestimmte Eisbudenstandorte stabil sind, kann an den Überlappungen der Bereiche erkannt werden. Wenn die Bereiche der E größten Überlappungen von unterschiedlichen Bereichen zusammen zu mehr als der Hälfte der Häuser gehören, dann bekommen neue Eisbudenstandorte, bei denen jede Eisbude auf eine der E höchsten Überlappungen gelegt wird, mehr als die Hälfte der Stimmen. Eisbudenstandorte sind also dann stabil, wenn die E höchsten Überlappungen unterschiedlicher Bereiche zusammen zu nicht mehr als der Hälfte der Häuser gehören.

Zwischen 2 Eisbuden gibt es höchstens 2 unterschiedliche Überlappungen, die alle Bereiche der Häuser enthalten, da jeder Bereich ein Ende an einer der 2 Eisbuden hat. Es muss nicht immer die Größe jeder einzelnen Überlappung gezählt werden, um zu bestimmen, ob bestimmte Eisbudenstandorte stabil sind. Wenn 2 der höchsten unterschiedlichen Überlappungen zwischen denselben Eisbuden sind, kann statt der Größe der Überlappungen die Anzahl der Häuser zwischen den Eisbuden gezählt werden, da jeder Bereich eines Hauses zu einem der beiden Überlappungen gehören muss.

Um eine Lösung zu finden, müssen zwischen allen benachbarten Eisbuden auf die Größe der größten Überlappung und die Anzahl der Häuser geachtet werden.

Im Folgenden wird mehrmals von Abständen gesprochen. 2 Punkte haben aber immer 2 verschiedene Abstände. Mit dem Abstand zweier Punkte ist von dem Punkt, der sich von der ersten Eisbude aus weniger gegen den Uhrzeigersinn befindet, der Abstand gegen den Uhrzeigersinn zum anderen Punk gemeint.

Die Größe der höchsten Überlappung zwischen 2 Eisbuden wird mithilfe von Gruppenbereichen reguliert. Diese können bestimmt werden, wenn mindestens eine Eisbude gegeben ist. Ein Gruppenbereich ist der Bereich einer Gruppe an benachbarten Häusern, in den die nächste Eisbude gestellt werden muss, damit sich nicht alle Bereiche der Gruppe überlappen.

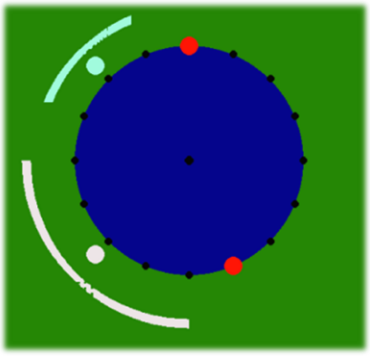
Nachdem die erste Eisbude platziert wurde, werden gegen den Uhrzeigersinn Gruppenbereiche durchgegangen, anhand denen nacheinander gegen den Uhrzeigersinn die nächsten Eisbuden platziert werden.

Alle Bereiche einer Gruppe überlappen sich dann nicht, wenn sich die Bereiche der beiden äußersten Häuser nicht überlappen. Damit sind von der ersten Eisbude aus gegen den Uhrzeigersinn das erste und letzte Haus der Gruppe gemeint.

Es gibt höchstens jeweils 2 Punkte, die als Anfang oder als Ende eines Gruppenbereiches infrage kommen. Das erste Haus kommt als Anfang und das letzte als Ende infrage, da deren Bereiche sich nicht überlappen, wenn eine Eisbude zwischen ihnen steht. Zum anderen kommt der Punkt als Ende infrage, der 1 mehr als doppelt so weit von der zuletzt platzierten Eisbude entfernt liegt wie der Abstand der beiden Häuser. Sollte dieser hinter dem letzten Haus der Gruppe liegen, ist der Abstand der letztplatzierten Eisbude zum ersten Haus zusammen mit dem Abstand des letzten Hauses zum Endpunkt 1 größer als der Abstand der Häuser. Wenn eine Eisbude auf den Endpunkt gestellt wird, überlappen sich die Bereiche der Häuser dann gerade so nicht. Der zweite Mögliche Anfangspunkt wird nicht gebraucht.

Wenn der Abstand der beiden äußeren Häuser mindestens halb so groß ist wie der Umfang des Sees, geht der Gruppenbereich um den ganzen See.

In diesem Bild wurde die zweite(untere) Eisbude so weit weg von der ersten(oberen) Eisbude gestellt wie möglich, wobei sich die Bereiche der beiden Häuser nicht überlappen. Der Abstand der beiden Häuser ist 4 Schritte groß und der Abstand der ersten und zweiten Eisbude ist 9 Schritte, also 1 mehr als doppelt so groß.



Es gibt verschiedene Varianten für die größtmöglichen Überlappungen und die Anzahl an Häusern zwischen den verschiedenen benachbarten Eisbuden. Welche Varianten es genau gibt und wie diese definiert sind, wird nun erklärt.

Jede Variante enthält zwischen allen 2 benachbarten Eisbuden die Größe der größtmöglichen Überlappung, kurz Höchstüberlappung. Die Höchstüberlappungen enthalten zusammen $floor(H/2)$ Bereiche. So gehören die Bereiche der E größten Überlappungen nicht zu mehr als der Hälfte der Häuser. Die Variante enthält außerdem eine Basis, welche der kleinsten Höchstüberlappung entspricht. Zwischen allen benachbarten Häusern darf es höchstens $Höchstüberlappung + Basis$ Häuser geben. Bei mehr Häusern wären die zwei unterschiedlichen Überlappungen zwischen den beiden Eisbuden größer als die Höchstüberlappung, zusammen mit einer anderen Höchstüberlappung, wodurch die größten Überlappungen mehr als $H/2$ Bereiche enthalten würden. Das heißt nicht, dass sich 2 der E größten Überlappungen nicht zwischen denselben Eisbuden befinden können, da nur von höchstmöglichen und nicht von tatsächlichen Überlappungen die Rede ist.

Um die verschiedenen Varianten zu bestimmen, wird nun der kleinstmögliche Wert, den eine Höchstüberlappung haben kann, bestimmt.

Es gibt E Paare an benachbarten Eisbuden. Zwischen jeder befinden sich höchstens 2 Überlappungen, die sich keine Bereiche teilen. Insgesamt gibt es also höchstens $E \bullet 2$ solche unterschiedlichen Überlappungen.

Es gibt mindestens $H - E$ Bereiche, da jedes Haus, bei dem keine Eisbude steht, genau einen Bereich hat. Von den Bereichen gehören $floor(H/2)$ zu den E höchsten Überlappungen. Mit einem Haus, bei dem eine Eisbude steht, ist gemeint, dass sich die Eisbude am Weg direkt gegenüber von dem Haus befindet. Die Bewohner solcher Häuser stimmen für keine neuen Eisbudenpositionen und haben somit keinen Bereich.

Wir bestimmen nun die Größen der kleinstmöglichen Überlappungen. Die Größte dieser ist nämlich genauso groß wie die kleinstmögliche Höchstüberlappung, da jede Höchstüberlappung mindestens so groß sein muss wie jede restliche Überlappung.

Die E höchsten Überlappungen enthalten zusammen $floor(H/2)$ Bereiche und die niedrigeren E Überlappungen enthalten die restlichen $H - E - floor(H/2)$ Bereiche. Die durchschnittliche Größe der niedrigeren E Überlappungen beträgt damit
$(H - E - floor(H/2))\ /\ E \mathrel{*}= (ceil(H/2) - E)/E = ceil(H/2)/E - 1$ Bereiche.
Der höchste Wert von den niedrigsten E Überlappungen und damit der Wert der kleinstmöglichen Höchstüberlappung ist dann $ceil(ceil(H/2)/E)\ -\ 1$.

*Es gilt $floor(H/2) + ceil(H/2) = H$ und somit $ceil(H/2) = H - floor(H/2)$

Die Basis einer Variante entspricht der kleinsten Höchstüberlappung. Sie hat meistens denselben Wert wie die kleinstmögliche Höchstüberlappung, kann aber auch höher sein.

Die Varianten enthalten alle Möglichkeiten für Verteilungen der Höchstüberlappungen. Dabei gilt, dass keine Höchstüberlappung kleiner als der eben bestimmte Wert ist und alle Höchstüberlappungen zusammen $floor(H/2)$ betragen.

Der Algorithmus, mit dem die Varianten erstellt werden, ist rekursiv. Zu Beginn werden alle möglichen ersten Höchstüberlappungen erstellt. Der Mindestwert für diese ist $ceil(ceil(H/2)/E) - 1$ und der Höchstwert ist $floor(H/2)$ subtrahiert mit dem Mindestwert jeder weiteren Höchstüberlappung. Für jede dieser Varianten wird der Prozess mit den verbleibenden Höchstüberlappungen wiederholt, bis alle Höchstüberlappungen bestimmt sind. Der Mindestwert bleibt gleich und beim Höchstwert werden zusätzlich die jeweiligen vorherigen Höchstüberlappungen abgezogen. Bei der letzten Höchstüberlappung wird nur der Höchstwert hinzugefügt, damit die Höchstüberlappungen zusammen $floor(H/2)$ groß sind.

Wenn es eine oder mehrere Lösungen gibt, dann steht bei mindestens einer davon mindestens eine Eisbude direkt an einem Haus. Warum das so ist, wird im Folgenden erklärt.

Wenn es stabile Eisbudenstandorte gibt, bei der keine Eisbude direkt an einem Haus steht, können alle Eisbuden gleichmäßig in eine Richtung verschoben werden, bis eine Eisbude direkt an einem Haus steht. 2 Häuser, dessen Bereiche sich nicht überlappen, weil eine Eisbude zwischen ihnen steht, werden sich weiterhin nicht überlappen, da sich daran nichts ändert. Häuser, dessen Bereiche sich nicht überlappen, da der Abstand der beiden nächsten Eisbuden höchstens 1 mehr als doppelt so groß ist wie der Abstand der Häuser, werden sich weiterhin nicht überlappen, da die nächsten Eisbuden dieselben bleiben und sich an beiden Abständen nichts ändert.

Da es keine Bereiche gibt, die sich überlappen, die das vorher noch nicht taten, sind die neuen Eisbudenstandorte, bei der eine Eisbude direkt an einem Haus steht, auch stabil.

Nun wird der Algorithmus vollständig erklärt.

Die erste Eisbude wird einmal direkt an jedes Haus gestellt. Für jeden solchen Durchgang werden alle Varianten durchgegangen und die anderen Eisbuden werden anhand dieser platziert.

Die zweite Eisbude wird anhand der ersten Höchstüberlappung der jeweiligen Variante platziert. Von den $Höchstüberlappung + Basis$ Häusern nach der ersten Eisbude werden die Enden der Gruppenbereiche aller Gruppen von benachbarten Häusern, die $Höchstüberlappung + 1$ groß sind, bestimmt.

Das letzte Haus der ersten Gruppe befindet sich $Höchstüberlappung + 1$ Häuser hinter der ersten Eisbude. Dies ist der nächste Punkt, an dem ein Ende eines der Gruppenbereiche sein kann. Das erste Haus der letzten Gruppe befindet sich $Basis$ Häuser hinter der ersten Eisbude. Dies ist der entfernteste Punkt, an dem ein Anfang eines der Gruppenbereiche sein kann. Da der entfernteste Anfang eines Gruppenbereichs immer vor dem nächsten Ende ist, überlappen sich immer alle Gruppenbereiche und die Anfänge dieser können ignoriert werden. Das nächste Ende eines der Gruppenbereiche ist der Punkt, der am weitesten von der ersten Eisbude entfernt liegt und an dem sich immer alle bestimmten Gruppenbereiche überlappen.

Wenn sich zwischen der ersten Eisbude und dem ersten Ende eines der Gruppenbereiche nicht mehr als $Höchstüberlappung + Basis$ Häuser befinden, wird die zweite Eisbude an diesem Punkt platziert und ansonsten an das Haus, dass $Höchstüberlappung + Basis + 1$ Häuser von der ersten Eisbude entfernt liegt.

Da die zweite Eisbude in allen bestimmten Gruppenbereichen liegt, kann sich keine Gruppe größer als die Höchstüberlappung zwischen den beiden Eisbuden Überlappen. Es wird auch sichergestellt, dass nicht mehr als $Höchstüberlappung + Basis$ Häuser zwischen ihnen liegen und dass die zweite Eisbude dabei die größtmögliche Entfernung zur ersten Eisbude hat.

Dieser Prozess wird immer mit der nächsten Höchstüberlappung der Variante und den $Höchstüberlappung + Basis$ Häusern nach der zuletzt platzierten Eisbude weitere $E - 1$-mal wiederholt.

Der Algorithmus hat damit für ein konstantes E, wie in der Aufgabenstellung gefragt, eine Laufzeit von $O(n^2)$, wobei die Eingabegröße n die Anzahl der Häuser ist. Die Anzahl der Varianten hat für ein gleichbleibendes E eine durchschnittliche Konstante und kann deshalb für die Laufzeit ignoriert werden.

Die letzte $(E + 1)$ Eisbude dient nur zum Test, ob die Eisbudenstandorte stabil sind, und wird am Ende gelöscht. Wenn die letzte Eisbude sich hinter oder auf der ersten Eisbude bzw. von der ersten Eisbude aus vor der vorletzten Eisbude befindet, sind die Eisbudenstandorte stabil. Wenn sich zwischen der vorletzten und letzten Eisbude nicht mehr als $Höchstüberlappung + Basis$ Häuser befinden und die höchste Überlappung nicht mehr als Höchstüberlappung Bereiche enthält ist, dann ist das zwischen der vorletzten Eisbude und der näher gelegenen ersten Eisbude auch der Fall.

Wenn in allen Durchgängen keine stabilen Eisbudenstandorte gefunden wurden, gibt es keine stabilen Eisbudenstandorte.

Umsetzung

Die Lösungsidee wird in Python implementiert.

Die kleinstmögliche Höchstüberlappung wird mit der Funktion math.ceil aus dem Modul math erstellt und in der Variable “minHöchstüberlappung” gespeichert.

Die benötigten Gruppen an Häusern werden in der Liste “Gruppensammlung” gespeichert. Diese enthält Listen mit jeweils allen Gruppen benachbarter Häuser einer Gruppengröße, die mit der Funktion “gruppe” erstellt werden. Die Gruppengrößen gehen von der kleinstmöglichen Höchstüberlappung + 1 bis zur größtmöglichen Höchstüberlappung + 1. Jede Gruppe ist eine Liste, in der die Hausnummer des letzten Hauses und 1 mehr als die doppelte Distanz der äußeren Häuser stehen. Damit kann jedes Ende eines Gruppenbereiches bestimmt werden.

Der Index jeder Gruppengröße in “Gruppensammlung” lässt sich bestimmen durch die Gruppengröße subtrahiert mit der kleinstmöglichen Höchstüberlappung und 1.

Alle Varianten werden in der Liste “Varianten” gespeichert und durch die Funktion “varianten” erstellt. Jede Variante ist eine Liste aus $E + 1$ Zahlen. Die ersten E sind die Höchstüberlappungen nach der jeweiligen Eisbude und die letzte Zahl ist die Basis.

Das Ende von jedem Gruppenbereich wird mit der Funktion „bereich“ bestimmt und in der Variable “BereichEnde” gespeichert. In der Funktion „bereich“ wird unter anderem die Funktion „verschieben“ aufgerufen, um den Punkt der 1 mehr als die doppelte Distanz der beiden Häuser von der zuletzt platzierten Eisbude entfernt liegt, zu bekommen.

Das Ende eines Gruppenbereiches, dass sich am wenigsten gegen den Uhrzeigersinn befindet bzw. am nächsten ist, wird in der Variable “MinBereichEnde“ gespeichert.

Ob ein Punkt sich weniger oder weiter gegen den Uhrzeigersinn befindet als ein anderer, wird mit den Funktionen “minplus” und “maxplus” bestimmt. In denen werden die Entfernungen gegen den Uhrzeigersinn von der ersten Eisbude zu diesen Punkten mit der Funktion “dist” gemessen. Der Punkt, der weiter weg ist, befindet sich weiter gegen den Uhrzeigersinn.

Immer nachdem die Position einer Eisbude bestimmt und zur Liste “Eisbuden” hinzugefügt wurde, werden die Indizes der nächsten sich nicht zu überlappenden Gruppen bestimmt. Der erste und der letzte Index werden mit den Variablen “Start” und “Ende” gespeichert. Der Index einer Gruppe entspricht dem Index des ersten Hauses der Gruppe. “Start” ist immer der Index des Hauses hinter der zuletzt platzierten Eisbude, welches in dem Dictionary “nächstesHaus” gespeichert ist. Dieses Dictionary enthält das nächste Haus gegen den Uhrzeigersinn von allen Positionen aus. “Ende” ist der Index des Hauses, dass sich $Basis + 1$ Häuser hinter dem von “Start” befindet.

Das Moduls sys wird benutzt, damit das Programm mit der einzulesenden Datei und der Anzahl der Eisbuden als Parameter auf der Kommandozeile aufgerufen werden kann. Die Dateien befinden sich dabei im selben Ordner, wie die Programmdatei.

Beispiele

Wir rufen das Python-Programm mit den verschiedenen BWINF-Eingabedateien und mit verschiedenen Eisbudenanzahlen auf. Alle Dateien liegen im gleichen Ordner, wie die Programmdatei.

$ ./Eisbudendilemma.py eisbuden1.txt 3

Es gibt folgende Varianten: [[1, 1, 1, 1]]

Die zuerst gefundenen stabilen Eisbudenstandorte lauten: [2, 12, 15]

$ ./Eisbudendilemma.py eisbuden1.txt 2

Es gibt folgende Varianten: [[1, 2, 1], [2, 1, 1]]

Die zuerst gefundenen stabilen Eisbudenstandorte lauten: [2, 12]

$ ./Eisbudendilemma.py eisbuden2.txt 3

Es gibt folgende Varianten: [[2, 2, 3, 2], [2, 3, 2, 2], [3, 2, 2, 2]]

Es gibt keine stabilen Eisbudenstandorte

$ ./Eisbudendilemma.py eisbuden2.txt 5

Es gibt folgende Varianten: [[1, 1, 1, 1, 3, 1], [1, 1, 1, 2, 2, 1], [1, 1, 1, 3, 1, 1], [1, 1, 2, 1, 2, 1], [1, 1, 2, 2, 1, 1], [1, 1, 3, 1, 1, 1], [1, 2, 1, 1, 2, 1], [1, 2, 1, 2, 1, 1], [1, 2, 2, 1, 1, 1], [1, 3, 1, 1, 1, 1], [2, 1, 1, 1, 2, 1], [2, 1, 1, 2, 1, 1], [2, 1, 2, 1, 1, 1], [2, 2, 1, 1, 1, 1], [3, 1, 1, 1, 1, 1]]

Die zuerst gefundenen stabilen Eisbudenstandorte lauten: [6, 27, 37, 39, 46]

$ ./Eisbudendilemma.py eisbuden3.txt 3

Es gibt folgende Varianten: [[2, 2, 4, 2], [2, 3, 3, 2], [2, 4, 2, 2], [3, 2, 3, 2], [3, 3, 2, 2], [4, 2, 2, 2]]

Die zuerst gefundenen stabilen Eisbudenstandorte lauten: [17, 42, 1]

$ ./Eisbudendilemma.py eisbuden4.txt 3

Es gibt folgende Varianten: [[3, 3, 3, 3]]

Es gibt keine stabilen Eisbudenstandorte

$ ./Eisbudendilemma.py eisbuden5.txt 3

Es gibt folgende Varianten: [[3, 3, 6, 3], [3, 4, 5, 3], [3, 5, 4, 3], [3, 6, 3, 3], [4, 3, 5, 3], [4, 4, 4, 4], [4, 5, 3, 3], [5, 3, 4, 3], [5, 4, 3, 3], [6, 3, 3, 3]]

Die zuerst gefundenen stabilen Eisbudenstandorte lauten: [83, 130, 233]

$ ./Eisbudendilemma.py eisbuden6.txt 3

Es gibt folgende Varianten: [[5, 5, 8, 5], [5, 6, 7, 5], [5, 7, 6, 5], [5, 8, 5, 5], [6, 5, 7, 5], [6, 6, 6, 6], [6, 7, 5, 5], [7, 5, 6, 5], [7, 6, 5, 5], [8, 5, 5, 5]]

Es gibt keine stabilen Eisbudenstandorte

$ ./Eisbudendilemma.py eisbuden7.txt 3

Es gibt folgende Varianten: [[6, 6, 8, 6], [6, 7, 7, 6], [6, 8, 6, 6], [7, 6, 7, 6], [7, 7, 6, 6], [8, 6, 6, 6]]

Die zuerst gefundenen stabilen Eisbudenstandorte lauten: [114, 285, 420]

Quellcode

```
# gruppen gibt alle Gruppen einer Gruppengröße zurück
def gruppen(n):
    Gruppen = []
    for Index, erstesHaus in enumerate(Häuser):
        letztesHaus = Häuser[(Index + (n - 1)) % Anzahl]
        # Jede Gruppe wird gespeichert mit der Position des letzten Hauses der Gruppe
        # und den Abstand, den die zuletzt platzierte Eisbude zur nächsten Eisbude haben muss,
        # damit die Gruppe sich nicht überlappt
        Gruppen += [[letztesHaus, 2 * dist(erstesHaus, letztesHaus) + 1]]
    return Gruppen

# Die Funktion gruppenbereich gibt das Ende des Gruppenbereiches einer Gruppe zurück
def gruppenbereich(Gruppengröße, Index2, Eisbuden):
    Gruppen = Gruppensammlung[Gruppengröße]
    # BereichEnde wird zuerst auf das letzte Haus der Gruppe gesetzt
    BereichEnde = Gruppen[Index2][0]
    # Eisbudenabstand ist der Abstand, den die zuletzt platzierte Eisbude zur nächsten Eisbude haben muss,
    # damit die Gruppe sich nicht überlappt
    Eisbudenabstand = Gruppen[Index2][1]
    # Wenn sich eine Gruppe über die hälfte des Sees erstreckt,
    # erstreckt sich der Gruppenbereich um den ganzen See
    if Eisbudenabstand <= 0:
        BereichEnde = Eisbuden[0] - 0.5
    else:
        # BereichEnde ist entweder am letzten Haus der Gruppe
        # oder Eisbudenabstand von der zuletzt platzierten Eisbude entfernt,
        # Je nachdem, was weiter gegen den Uhrzeigersinn liegt
        BereichEnde = maxplus(
            Eisbuden[-1], BereichEnde, verschieben(Eisbuden[-1], Eisbudenabstand)
        )
    return BereichEnde

# Die rekursive Funktion varianten erstellt alle Varianten
# Variante ist eine Variante, die für jeden rekursiven Aufruf der Funktion anders ist
# Sie ist zuerst leer, bekommt aber mit jeden tieferem Aufruf eine Höchstüberlappung mehr
# n ist die gewollte Summe der noch fehlenden Höchstüberlappungen und entspricht zu Beginn H//2
# Anzahl ist die Anzahl der noch fehlenden Höchstüberlappungen und entsprich zu Beginn E
def varianten(Variante, n, Anzahl):
    # Die letzte Höchstüberlappung hat die Größe von n
    if Anzahl == 1:
        # Die Basis ist die kleinste Höchstüberlappung
        return [Variante + [n, min(Variante + [n])]]
    # Varianten enthält alle Varianten
    Varianten = []
    # Zu jeder Variante werden alle möglichen Höchstüberlappungen hinzugefügt
    # und für jede wird die Funktion rekursiv aufgerufen
    for i in range(minHöchstüberlappung, n - ((Anzahl - 1) * minHöchstüberlappung) + 1):
```

```
            Varianten += varianten(Variante + [i], n - i, Anzahl - 1)
    # Varianten wird ausgegeben
    return Varianten

# minHöchstüberlappung ist die kleinstmögliche Höchstüberlappung
minHöchstüberlappung = math.ceil(math.ceil(Anzahl / 2) / Eisbudenanzahl) - 1

# Gruppensammlung enthält alle Gruppen mit Gruppengrößen 1 Größer als mögliche Höchstüberlappungen
Gruppensammlung = []
for i in range(
    minHöchstüberlappung + 1,
    Anzahl // 2 - (Eisbudenanzahl - 1) * minHöchstüberlappung + 2,
):
    Gruppensammlung += [gruppen(i)]

# Die Varianten werden mit der Funktion varianten erstellt
Varianten = varianten([], Anzahl // 2, Eisbudenanzahl)
# Die Varianten werden ausgegeben
print("Es gibt folgende Varianten:", Varianten)

# nächstesHaus enthält zu jeder Position den Index des Hauses,
# dass sich als Nächstes gegen den Uhrzeigersinn befindet
nächstesHaus = {}
# Zuerst ist der Index 0
Index = 0
# Jede Schrittzahl wird durchgegangen
for i in range(0, Umfang):
    # Wenn die Schrittzahl das nächste Haus erreicht, wird der Index erhöht
    if i == Häuser[Index % Anzahl]:
        Index += 1
    # Der Index wird zur Position eingetragen
    nächstesHaus[i] = Index

# Die Funktion eisbudendilemma findet stabile Eisbudenstandorte
def eisbudendilemma():
    # Jedes Haus und Variante werden durchgegangen, die erste Eisbude wird auf das Haus gesetzt
    for Index, Haus in enumerate(Häuser):
        for Variante in Varianten:
            Eisbuden = [Haus]
            Basis = Variante[Eisbudenanzahl]
            # Start und Ende sind die Indices der ersten Häuser der ersten und letzten Gruppe,
            # dessen Gruppenbereiche nach der ersten Eisbude erstellt werden
            Start = Index + 1
            Ende = Start + Basis + 1
            # Es werden E Eisbuden gesetzt
            for a in range(0, Eisbudenanzahl):
                # Höchstüberlappung ist die Höchstüberlappung nach der letzten Eisbude, die platziert wurde
                Höchstüberlappung = Variante[a]
                # BereichEnde ist das Ende eines Bereiches, dass sich weiter gegen den Uhrzeigersinn befindet
```

```
                # MinBereichEnde ist das BereichEnde, das sich am wenigstem gegen den Uhrzeiger-
sinn befindet
                MinBereichEnde = Eisbuden[-1] - 0.5
                # Alle Gruppen mit der Größe Höchstüberlappung von den Höchstüberlappung + Ba-
sis Häusern
                # Nach der zuletzt platzierten Eisbude werden durchgegangen
                # Index2 ist der Index der jeweiligen Gruppe
                for Index2 in range(Start, Ende):
                    Index2 %= Anzahl
                    # Das Ende des Gruppenbereichs wird bestimmt
                    BereichEnde = gruppenbereich(
                        Höchstüberlappung - minHöchstüberlappung, Index2, Eisbuden
                    )
                    # Wenn BereichEnde sich weniger gegen den Uhrzeigersinn befindet
                    # als das bisherige MinBereichEnde, wird dieses nun angepasst
                    if dist(Eisbuden[-1], BereichEnde) < dist(
                        Eisbuden[-1], MinBereichEnde
                    ):
                        MinBereichEnde = BereichEnde
                # Die nächste Eisbude wird entweder beim MinbereichEnde oder am Haus,
                # dass Höchstüberlappung + Basis + 1 Häuser von der zuletzt platzierten Eisbude ent-
fernt liegt, platziert
                # Je nachdem, welcher Punkt weniger gegen den Uhrzeigersinn liegt
                Eisbuden += [
                    minplus(
                        Eisbuden[-1],
                        MinBereichEnde,
                        Häuser[(Start + Höchstüberlappung + Basis) % Anzahl],
                    )
                ]
                # Start ist der Index vom ersten Haus nach der eben platzierten Eisbude
                # Ende ist Basis + 1 Häuser dahinter
                Start = nächstesHaus[Eisbuden[-1]]
                Ende = Start + Basis + 1
            # Wenn die letzte Eisbude von der ersten Eisbude aus vor der vorletzten Eisbude liegt,
            # sind die Eisbudenpositionen stabil und sie werden ohne die letzte Eisbude ausgegeben
            if dist(Eisbuden[0], Eisbuden[-1]) < dist(Eisbuden[0], Eisbuden[-2]):
                return Eisbuden[:-1]

# Die Lösung wird ausgegeben
Eisbuden = eisbudendilemma()
if Eisbuden:
    print("Die zuerst gefundenen stabilen Eisbudenstandorte lauten:", Eisbuden)
else:
    print("Es gibt keine stabilen Eisbudenstandorte")
```