АНАЛИЗ НА ЗАДАЧАТА OUTFIT

***Тагове****: Greedy, STL/ Set, графи, двойкосъчетания*

**Подзадача 1**

Ограничението за $N$ позволява всякакви *brute force* решения. Решението, предложено в outfit\_1.cpp, проверява всички пермутации на числата от 1 до $N$, а това в outfit\_backtracking.cpp използва рекурсия.

Сложност: $О(N!)$

**Подзадача 2**

Представяме децата и екипите като двуделен граф, за който броят върхове е равен на $2N$, а ребрата отразяват желанията на децата (дете с номер $i$ e свързано с екип с номер $j$, ако $j$ попада в интервала на детето). Използваме алгоритъмa на Кун за намиране на максимално двойкосъчетание. Ако то е $N$, извеждаме решението. Реализацията се намира в outfit\_2.cpp.

Сложност: $О(N^{3})$

**Подзадача 3**

Нека се опитаме последователно да разпределим екипите между децата. Въпреки че в началото са налични доста възможности, след известно време може за някои деца да не остане нито един екип по вкуса му и да заключим, че няма решение, което би било грешно. Следователно тази *greedy* стратегия няма да е точна всеки път.

 Да разпределяме тогава екипите последователно. Нужно е за определен модел да разглеждаме само децата, в чиито интервали попада номера му. Когато числото попада само в един интервал, значи това е детето, което трябва да го вземе. Какво правим в ситуацията, когато повече от едно деца желаят дадения екип? Отново подхождаме алчно. Ще дадем екипа на детето, чийто интервал е с най – малка дясна граница. Тъй като не е ясно дали в последствие ще се намери друг екип, който да му допада, най – добре отрано да се подсигурим.

 Достатъчно условие да няма решение е за дадена последователност от екипи с дължина *p* броят на изцяло попадащите вътре интервали да е по – голям от *p*.

Нека в даден момент два или повече интервала с дясна граница *R* трябва да вземат *R*-тия модел. Ако се окаже, че все пак съществува решение, то алгоритъмът ще е грешен. Нека най – малкият ляв край на гореспоменатите интервали е $lmin$. За всеки един екип от $lmin$ до *R-1* е имало поне по едно дете, чиято дясна граница е била по – малка от *R*. Нека „преместим“ $lmin$ на минималната ляв край на тези интервали. В рамките на $lmin$ и *R* интервалите или предизвикват преместването на $lmin$ или се затварят. Нека крайната стойност на$ lmin$ да е *L.* Следователно между *L* и *R-1* има *R-L* затворени интервала. Но на позиция *R* се затварят още поне два, което води до общ брой изцяло попадащи вътре интервали поне *R-L+2*. Но $R-L+2>R-L+1$, значи няма решение.

Обхождаме числата (екипите) от 1 до $N$. Достатъчно е да поддържаме множество от отворени интервали на деца, за които все още не сме избрали екип. Избираме от множеството този интервал, чийто десен край е най – малък и записваме текущото число от итерацията. Ако множеството е празно в даден момент или десният край на избрания интервал е по – малък от номера на текущия екип, то няма решение. Сортираме интервалите по ляв край, за да може да ги вкараме по-лесно в множеството при прехода от $x$ към $x+1$ (трябва да включим тези с ляв край $x+1$). Вградените структури $set $(outfit\_full.cpp) и $priority\\_queue$ са много удобни за целта.

Сложност: $О(NlogN)$

 *Условие, решние, тестове и анализ: Дениз Потурлиев*