Hyperloop I

Termín odevzdání:	**25.11.2018 23:59:59**
Pozdní odevzdání s penalizací:	**06.01.2019 23:59:59** (Penále za pozdní odevzdání: 100.0000 %)
Hodnocení:	**0.1875**
Max. hodnocení:	**3.0000** (bez bonusů)
Odevzdaná řešení:	1 / 20 Volné pokusy + 10 Penalizované pokusy (-10 % penalizace za každé odevzdání)
Nápovědy:	0 / 2 Volné nápovědy + 2 Penalizované nápovědy (-10 % penalizace za každou nápovědu)

Úkolem je realizovat funkci (ne celý program, pouze funkci), která bude navrhovat hyperloop tratě. Funkce bude na základě parametrů určovat, jak poskládat segmenty tratě, aby vznikla trať přesně požadované délky.

Trať pro hyperloop je tvořena tubusem kruhového průřezu. Tubus je složen se segmentů určité délky. Předpokládáme, že segmenty dodávají dva výrobci, každý výrobce dodává segment s jednou fixní délkou. Dvojice sousedních segmentů je spojena právě jednou vzduchotěsnou přepážkou. Dále, vzduchotěsnou přepážkou musí celý tubus začínat i končit. Všechny přepážky mají stejnou délku. Segmenty ani přepážky nelze zkracovat ani prodlužovat. Úkolem funkce je určit, zda lze sestavit ze segmentů a přepážek tubus přesně zadané délky. Pokud tubus zadané délky sestavit lze, funkce navíc určí i počty potřebných segmentů jednotlivých délek a určí, kolika různými způsoby lze tubus sestavit.

Platné sestavení tubusu je tedy:

```
bulkhead
bulkhead segment bulkhead
bulkhead segment bulkhead segment bulkhead
bulkhead segment bulkhead segment bulkhead segment bulkhead
bulkhead segment bulkhead segment bulkhead segment bulkhead segment bulkhead
...
```

Požadovaná funkce má následující rozhraní:

```
unsigned long long int  hyperloop                       ( unsigned long long int len,
                                                          unsigned int s1,
                                                          unsigned int s2,
                                                          unsigned int bulkhead,
                                                          unsigned int * c1,
                                                          unsigned int * c2 );
```

`len`
: je vstupní parametr udávající požadovanou délku tubusu,

`s1, s2`
: jsou vstupní parametry udávající délky segmentu od prvního a druhého výrobce. Jeden nebo oba parametry mohou být zadané jako 0, tato hodnota znamená, že segmenty od daného výrobce nejsou k dispozici (při konstrukci nesmí být použity).

`bulkhead`
: je vstupní parametr udávající délku spojovací vzduchotěsné přepážky. Parametr může být zadaný jako 0, přepážky pak neprodlouží celkovou délku tubusu,

`c1, c2`
: jsou výstupní parametry, které udávají počet potřebných segmentů od prvního/druhého výrobce. Funkce při výpočtu nastaví tyto parametry na hodnotu libovolného jednoho nalezeného řešení. Pokud se funkci nepodaří najít žádné řešení, ponechá oba parametry beze změn,

návratová hodnota

: návratovou hodnotou funkce je počet nalezených řešení, případně informace o neúspěchu. Pokud výpočet neuspěje (požadované délky nelze dosáhnout žádnou platnou kombinací segmentů a přepážek) vrátí funkce hodnotu 0. Pokud požadované délky dosáhnout lze, funkce vrátí počet existujících různých kombinací segmentů, které vedou k požadované délce. Výstupní parametry `c1` a `c2` nastaví na libovolné jedno z těchto řešení.

Dvě řešení považujeme za různá, pokud se v nich liší požadované počty segmentů nějaké délky. Naopak, pokud obě řešení vyžadují stejný počet segmentů v každé délce, považujeme je za ekvivalentní a do výsledku je započteme pouze jednou. Tedy:

- změna pořadí segmentů v tubusu nehraje roli. Například pro:

  ```
  len = 20
  s1 = 5
  s2 = 10
  bulkhead = 0
  ```

  existují možnosti:

  ```
  [bulkhead] [segment 5] [bulkhead] [segment 5] [bulkhead] [segment 5] [bulkhead] [segment 5] [bulkhead]

  [bulkhead] [segment 5] [bulkhead] [segment 5] [bulkhead] [segment 10] [bulkhead]
  [bulkhead] [segment 5] [bulkhead] [segment 10] [bulkhead] [segment 5] [bulkhead]
  [bulkhead] [segment 10] [bulkhead] [segment 5] [bulkhead] [segment 5] [bulkhead]
  ```

```
[bulkhead] [segment 10] [bulkhead] [segment 10] [bulkhead]
```

Prostřední tři možnosti se liší pouze pořadím segmentů, ale všechna požadují jeden segment délky 10 a dva segmenty délky 5. Proto prostřední tři možnosti považujeme za ekvivalentní, funkce bude pro tento vstup vracet výsledek 3:

```
4 * s1 + 0 * s2 + 5 * bulkhead = 20
2 * s1 + 1 * s2 + 4 * bulkhead = 20
0 * s1 + 2 * s2 + 3 * bulkhead = 20

hyperloop ( 20, 5, 10, 0, &c1, &c2) == 3
```

- pokud oba výrobci vyrábějí segmenty stejné délky, pak jsou tyto segmenty neodlišitelné. Například pro:

```
len = 40
s1 = 10
s2 = 10
bulkhead = 0
```

existuje 5 možných poskládání segmentů do výsledku:

```
4 * s1 + 0 * s2 + 5 * bulkhead = 40
3 * s1 + 1 * s2 + 5 * bulkhead = 40
2 * s1 + 2 * s2 + 5 * bulkhead = 40
1 * s1 + 3 * s2 + 5 * bulkhead = 40
0 * s1 + 4 * s2 + 5 * bulkhead = 40
```

Těchto 5 možností ale považujeme za ekvivalentní, protože každé z řešení požaduje 4 kusy segmentů délky 10.

```
hyperloop ( 40, 10, 10, 0, &c1, &c2 ) == 1
```

Odevzdávejte zdrojový soubor, který obsahuje implementaci požadované funkce `hyperloop`. Do zdrojového souboru přidejte i další Vaše podpůrné funkce, které jsou z ní volané. Funkce `hyperloop` bude volaná z testovacího prostředí, je proto důležité přesně dodržet zadané rozhraní funkce. Za základ pro implementaci použijte kód z přiloženého souboru. V kódu chybí vyplnit tělo požadované funkce (a případné další podpůrné funkce). Ukázka obsahuje testovací funkci `main`, uvedené hodnoty jsou použité při základním testu. Všimněte si, že vkládání hlavičkových souborů a funkce `main` jsou zabalené v bloku podmíněného překladu (`#ifdef/#endif`). Prosím, ponechte bloky podmíněného překladu i v odevzdávaném zdrojovém souboru. Podmíněný překlad Vám zjednoduší práci. Při kompilaci na Vašem počítači můžete program normálně spouštět a testovat. Při kompilaci na Progtestu funkce `main` a vkládání hlavičkových souborů "zmizí", tedy nebude kolidovat s hlavičkovými soubory a funkcí `main` testovacího prostředí.

Váš program bude spouštěn v omezeném testovacím prostředí. Je omezen dobou běhu (limit je vidět v logu referenčního řešení) a dále je omezena i velikost dostupné paměti. Rozumná implementace naivního algoritmu by měla projít všemi testy kromě testu rychlosti. Pro zvládnutí testu rychlosti je potřeba použít výkonnější algoritmus, který zbytečně netestuje všechny možné kombinace segmentů.

Nápověda:

- Jako základ Vašeho řešení použijte zdrojový kód z přiloženého souboru.
- Do funkce `main` si můžete doplnit i další Vaše testy, případně ji můžete libovolně změnit. Důležité je zachovat podmíněný překlad.
- V ukázce je použito makro `assert`. Pokud je parametrem nenulová hodnota, makro nedělá nic. Pokud je parametrem nepravda (nula), makro ukončí program a vypíše řádku, kde k selhání došlo. Pokud tedy Vaše implementace projde ukázkovými testy, program doběhne a nic nezobrazí.
- Požadovaná délka tubusu a počet existujících řešení může být větší než rozsah datového typu `int`. Parametry jsou proto v rozhraní deklarované jako `unsigned long long int`, tedy s velikostí 64 bitů. Pro čtení a zobrazení těchto hodnot použijte konverzi `%llu`.
- Nesnažte se vygenerovat všechna řešení a následně z nich odstraňovat ekvivalentní řešení. Generujte řešení rovnou bez duplicit. Úloha k vyřešení nevyžaduje pole. Použití pole (zde by dokonce muselo být dynamicky alokované) je zbytečně pracné, odstraňování duplicit je zbytečně časově náročné.
- Datový typ `unsigned long long int` je rozšířením, které není součástí normy jazyka C90/C++03. Při kompilaci podle těchto standardů kompilátor hlásí varování. Kompilátor na Progtestu je nastaven tak, že tato varování potlačuje (je použit přepínač `-Wno-long-long`).

Vzorová data: **Download**

☐ **Referenční řešení**

1	**25.11.2018 22:15:45**	**Download**
Stav odevzdání:	Ohodnoceno	
Hodnocení:	0.1875	

- **Hodnotitel: automat**
  - Program zkompilován
  - Test 'Zakladni test podle ukazky': Úspěch
    - Dosaženo: 100.00 %, požadováno: 100.00 %
    - Celková doba běhu: 0.895 s (limit: 1.000 s)

        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Test 'Test meznich hodnot': Program překročil přidělenou maximální dobu běhu
        - Vyčerpání limitu na celý test, program násilně ukončen po: 2.003 s (limit: 2.000 s)
        - Neúspěch v nepovinném testu, hodnocení: 25.00 %
    - Test 'Test nahodnymi daty': Nebylo testováno
        - Neúspěch v nepovinném testu, hodnocení: 25.00 %
    - Test 'Bonusovy test (rychlost)': Nebylo testováno
        - Neúspěch v bonusovém testu, hodnocení: Bonus nebude udělen
    - Celkové hodnocení: 6.25 % (= 1.00 * 0.25 * 0.25)
- Celkové procentní hodnocení: 6.25 %
- Celkem bodů: 0.06 * 3.00 = 0.19

		Celkem	Průměr	Maximum	Jméno funkce
SW metriky:	Funkce:	**2**	--	--	--
	Řádek kódu:	**133**	**66.50 ± 43.50**	**110**	`hyperloop`
	Cyklomatická složitost:	**39**	**19.50 ± 8.50**	**28**	`hyperloop`