Кръг III. Tower defense

Мечо Пух е много запален геймър. Някак странно, но факт – до скоро той не подозираше за tower defense игрите! „Как не съм знаел за тях до сега!?“ - помисли си той. Тъй като Мечо Пух не е много опитен, той често не стига до последната вълна и това страшно го натъжава. Вашата задача в този кръг ще е да напишете програма, която показва на Мечо Пух как да достигне до вълната с най-голям възможен номер!

Tower defense-а, който той играе в момента може да се опише накратко по този начин:

1. Картата е поле N x M (N, M <= 64)
2. Всяко поле от картата е някое от следните:
   - Блокирано (на него не може да строи)
   - Начало, край или междинно поле от пътя на чудовищата
     Пътят винаги е еднозначен и може да се определи по следния алгоритъм:

     За всяко поле от пътеката (започвайки от ‘S’), с изключение на ‘E’, винаги ще съществува уникално поле ‘#’ или ‘E’, такова, че то не е това, от което сме дошли. Пътеката продължава само по 4те посоки (нагоре, надолу, наляво, надясно).

   - Свободно за строеж на кули поле
3. Съществуват различни видове чудовища, като всяко чудовище е от точно един тип.
4. Съществуват три типа кули:
   - Биещи кули, които могат да нанасят щета едновременно на много видове чудовища (но не задължително всички)
   - Модифициращи кули за щета
   - Модифициращи кули за бързина

Възможно е дадена кула да изисква няколко други кули да бъдат вече построени. Кулите от които дадена кула зависи винаги се премахват, за да може да се построи новата.

5. Едновременно могат да бъдат построени максимум 64 кули.
6. Мечо Пух може да прави 4 операции между вълните:
   - Да купува кула на дадени координати
   - Да продава кула на дадени координати

- Да променя *максималната загубена щета* (за пояснение, моля прочетете надолу)
- Да определи подредбата, в която кулите ще бият по чудовища от даден тип или да каже, че иска да използва тази по подразбиране (първо бият кулите, които нанасят най-много щета)

  В началото всички кули използват подредбата по подразбиране.

  Ако е зададена подредба и бъдат закупени нови кули, то единственото гарантирано нещо е, че новите кули ще бият винаги след кулите зададени в подредбата (но не задължително в подребата, в която са закупени)

7. Пари се получават не на убито животно, а на мината вълна. Във входният файл е зададен и броят на началните пари.
8. Всяко вълна завършва, когато за всяко от животните е вярно, че:
   - Е убито
   - Или е стигнало до края
9. Всяко животно притежава три характеристики – бързина, тип, живот.

   Бързината е в единици **полета/секунда**. Всяка вълна продължава няколко секунди (докато условията в т. 8 не бъдат изпълнени). През всяка секунда, животното се предвижва съответният брой единици в посока края на пътеката.
10. Всяка кула притежава следните характеристики:
    - Обсег, който описва **квадрат** (не окръжност!) с център самата кула.
    - Щета, която кулата нанася на животните от всеки тип ИЛИ (ако е модифицираща кула) с колко кулата променя щетата/бързината на кулите в обсег (Например, за 5% намаление на бързината: -0.05)
    - Име (неуникално, служи просто за дебъг)
    - Бързина, изразяваща колко пъти за една секунда ще стреля кулата, ако е положително число. Ако е отрицателно число **-r**, то кулата ще бие по всяко животно във обсега по **r** пъти на секунда (без значение от броя на животните, mass damage☺)
    - Цена за закупуване и пари, получени при продажба.

По-детайлна информация ще бъде предоставена в отделен файл.

Вход:

От първия ред на входния файл **towerdef.in** се въвеждат две цели числа – N, M.

На следващите N реда се въвежда по M знака, със следните значения:

1. ‘S’ – начало на пътеката, по която се движат животните. Всички животни започват вълните от тук.

2. ‘E’ – край на пътеката. Животните трябва да минат това поле, т.е. ако просто го достигнат, не са достигнали изхода.

3. ‘#’ – междинно поле от пътеката.

4. ‘.’ – свободно за строеж поле.

5. ‘x’ – блокирано поле.

От следващия ред се въвежда цяло число **T**, означаващо броя на кулите, които могат да бъдат строени (първата е кула №0, втората №1, …). Следват съответните **T** описания на кули, всяко от които във формата:

1. Име на кулата, никога не съдържа интервали.
2. Тип на кулата (цяло число, 2 за биеща, 1 за модифицираща щета, 0 за модифицираща бързина)
3. Бързина в брой стреляния / секунда (както бе обяснено, може да е отрицателно).
4. Обсег на кулата. Описва квадрат, с център позицията на кулата.
5. Цена за закупуване
6. Пари, върнати при продажба на кулата.
7. Следват 16 реални числа, описващи щетата, която кулата нанася на всеки от 16те вида чудовища.
8. Следва цяло число **U**, описващо броя на кулите, които трябва да бъдат предоставени, за да бъде построена кулата. Следват **U** номера (0 индексирани, както предоставени във входния файл) на съответните кули. Кула никога няма да зависи от себе си (директно или индиректно), за да бъде построена.

От следващият ред се въвежда цяло число **M**, показващо колко вида животни има. Всеки от следващите M реда описва по едно животно, в следния формат:

1. Тип на животното (между 0 и 15, за всеки от 16те типа животни)
2. Бързина в полета/секунда. Цяло число.
3. Реално число – точки живот. Животните винаги започват с максимален живот всяка вълна.

Следва ред с две цели числа – брой на животите и начални пари. Началните пари винаги ще се побират в 64 бита, но не задължително в 32 бита.

Следва цяло число **W** – броят на вълните. Следват W описания на вълни 1..W, всяко във формат:

1. Две числа – цяло число **P** и 64 битово число – пари, които ще бъдат получени, ако след завършването на тази вълна, останалите животи са повече от 0.
2. Следват **P** двойки цели числа. Първото – указващо тип на животните, а второто – броя им. Никога няма да има две двойки с еднакъв тип на животните (в рамките на една вълна).

Изход:

На първия ред от изходния файл **towerdef.out** вашата програма трябва да изведе инструкциите, които позволяват на Мечо Пух да победи всички вълни. Формата е както следва:

За всяка вълна от 1 до W:

1. Първо цяло число **I** – броя на инструкциите, които да бъдат изпълнени преди вълната.
2. Следват I описания на инструкции, всяка от които може да е:
    1. Да се купи кула (4)
    2. Да се продаде кула (5)
    3. Да се промени подредба на стрелба (1)
    4. Да се промени максималната загубена щета (3)

    Всяка инструкция започва с цяло число, описващо типа на инструкцията – 1 за промяна на подребата, 3 – за максимална загубена щета, 4 – за закупуване на кула и 5 – за продажба.

    Ако се променя подредба, инструкцията би изглеждала по този начин: *1 <тип на животните> x0 y0 x1 y1 x2 y2 ….*

    Където *тип на животните* показва, че тази подреба важи, когато кулите бият по животно от съответния тип. Числата *xi yi* за I от 0 до *броя на построените кули* – 1 указва подредбата. Всяка кула трябва да присъства точно 1 път.

    Задаване на подредба може да се провали, ако не са изброени всички построени кули или някоя не е спомената.

    Ако се променя максималната загубена щета, то инструкцията би изглеждала по този начин:

3 <ред> <колона> <максимална загубена щета>

Ред и колона са дефинирани по обичайния начин. Те трябва да са координатите на зает с кула поле. Ако са зададени невалидни координати или полето е част от пътя на чудовищата или е блокирано или е свободно, то изпълнението на инструкцията ще се провали. Максималната щета е реално число, което лимитира какво става, когато една кула би направила щета X на чудовище, с живот Y, където X >= Y. Загубената щета дефинираме като (X – Y)/X и ако това число е по-голямо от *максимална загубена щета*, то кулата няма да стреля. По подразбиране, кулите ще стрелят винаги (т.е. максималната загубена щета е 1.0)

Ако се закупува кула, инструкцията би изглеждала по този начин:

4 <ред, на който да бъде поставена кулата> <колона> <номер на кулата, число от 0 до T – 1 (вж. Вход)> x0 y0 x1 y1 …

Двойките *xi yi* ще са точно толкова, от колкото кули зависи кулата, която искате да закупите. Всяка двойка *xi yi* трябва да е валидна локация на кула към момента на изпълнение на инструкцията и трябва да е точно от типа, който е съответния номер нужна кула, указана във входа. (Пример: ако кула №3 изисква кули №0 и №1, инструкцията за построяването и на (11,12) би била:

4 11 12 3 0 1 2 3

Като се предполага, че на (0,1) има кула от тип №0, а на (2,3) – от тип №1. Тези кули ще бъдат премахнати преди закупуването на новата кула, така че локацията на новата кула може да съвпада с някоя от техните локации (т.е. вместо (11,12), тя може да е (0,1) или (2,3)).

Закупуване на кула може да се провали, ако:

- Номерът на кулата е невалиден (например, отрицателен)
- Координатите са невалидни или полето, което идентифицират, не е свободно за строеж (след премахването на кулите, които тази кула изисква за своето построяване)
- Координатите на някоя нужна кула са невалидни или не са координати на кула от правилния тип.
- **Опитвате се да построите повече от 64 кули**

Ако се продава кула, инструкцията би изглеждала по този начин:

5 <ред> <колона>

На (ред,колона) трябва да има построена кула. Ще бъдат получени пари, спрямо данните, предоставени във входа. Продажба се проваля единствено, когато координатите са невалидни или на тях няма кула.

Ако някоя инструкция се провали получавате 0 точки. Трябва да се изпишат 0 дори и да не правите нищо от вълна X нататък до края (т.е. ако само закупите кула преди вълна №1 от 20, трябва да имате 19 нули след това, указващи, че не правите нищо). Ако не спазвате това правило, отново ще получите 0.

Пример 1:

towerdef.in	towerdef.out:	Обяснение:
2 5 S###E ..... 2 RocketLauncher 2 5 3 100 60 2.0 2.0 2.0 2.0 0 0 0 0 0 0 0 0 0 0 0 0 0 Turret2 2 5 3 100 60 3.0 1.0 2.0 2.0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 7 10 200 2 1 200 0 15 1 300 0 7	1 4 1 3 0 2 4 1 2 1 1 0 1 3 1 2	**На входа:** Дадена е карта 2x5, като животните се движат от ляво на дясно по първия ред. Има два вида кули – „RocketLauncher” и „Turret2” Има 1 животно от тип 0, с бързина 1 и живот 7. Началните животи са 10, а пари – 200. Има 2 вълни. Първата вълна съдържа 15 животни от тип 0, а втората – 7. **На изхода:** Преди вълна 1 – купи кула от тип №0(„RocketLauncher”) и я постави на (1,3) (индексирането е от 0). Преди вълна 2 – купи кула от тип №1 („Turret2”) и я постави на (1,2). След това кажи за животни от тип0, първо да стреля кулата на (1,3), а след това – тази на (1,2). По този начин и двете вълни са минати.

Пример 2:

towerdef.in	towerdef.out:	Обяснение:

2 5 S###E ..... 2 RocketLauncher 2 5 3 100 60 2.0 2.0 2.0 2.0 0 0 0 0 0 0 0 0 0 0 0 0 0 Turret2 2 -1 3 120 60 7.0 1.0 2.0 2.0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 7 10 220 2 1 200 0 15 1 300 0 7	1 4 1 2 0 1 4 1 2 1 1 2	**На входа:** Разликата е, че вече Turret2 е ъпгрейд на RocketLauncher и прави mass damage (бързина = -1). **На изхода:** Преди вълна 1 – купи кула от тип №0(„RocketLauncher”) и я постави на (1,2) (индескирането е от 0). Преди вълна 2 – купи кула от тип №1 („Turret2”) и я постави на (1,2) (тъй като ъпгрейдваме кулата на (1,2), то полето е свободно за строеж). През втората вълна никое животно не остава живо ☺

Оценяване:

По формулата $\left(\frac{\mathbf{yours + 1}}{\mathbf{max + 1}}\right)^{\mathbf{3}} \times \frac{\mathbf{100}}{\mathbf{T}}$, където T е броят на тестовете, yours – достигната от вас вълна, а max – най-голямата вълна, достигната от някой участник.

Чекър:

Ще бъде качен C++ симулатор на процеса, с който ще можете да тествате своите .out файлове. Всички официални тествания ще бъдат правени със същата програма. Тъй като проекта е голям и е съставен от няколко файла, за да го компилирате (ако никога не сте компилирали нещо по-сложно от 1 файл):

Обяснението е за Code::Blocks, но процеса би трябвало да е подобен във всяко IDE.

Правите нов проект и добавяте всички .cpp и .hpp файлове към него. Давате Build > Rebuild. След като всичко завърши (би трябвало без грешки ☺), давате Build and Run (F9). Можете да използвате кода във вашето решение. За информация и обяснения по самия код, ще има качен отделен файл.

Ограничения:

2 ≤ **N, M** ≤ 64

максимален брой кули, построени едновременно = 64

максимален брой различни типове животни = 16. Навсякъде, където се иска тип на животно се изисква цяло число между 0 и 15.

Максимална дължина на името на кула = 32.

Максимален брой инструкции преди всяка вълна = 1024.

Началните животи винаги ще се побират в 32 битово число със знак.

Началните пари ще се побират в 64 битово число без знак.

Няма да има повече от 3000 вълни.