ПРОЈЕКАТ

Из објектно-орјентисаног програмирања

ИГРА “МОЈ БРОЈ”

Студент:

Теодор Видаковић SV 33/2021

3.1.2023. година

У/И подсистем

На почетку свако рунде исписују се понуђени бројеви и тражени број. Затим један од такмичара уноси свој израз чији резултат провјерава рачунар. Ако је израз валидан, други такмичар уноси свој израз након чега се упоређује удаљеност од траженог броја. У случају да је један од играча погодио тачан број, рунда се аутоматски завршава и почиње наредна.

```
////////////////////////// Runda 1 ////////////////////////////
Trazeni broj : 456

Ponudjeni brojevi: 1  5  8  9  10  25
Igrac A na potezu:
(25-1)*(10+9)
 = 456
Igrac A je nasao tacan broj.


 Racunar: (25-1)*(9+10) = 456

 -----> Ako zelite zavrsiti igru unesite Q,ostalo za nastavak: s

////////////////////////// Runda 2 ////////////////////////////
Trazeni broj : 397

Ponudjeni brojevi: 3  3  5  9  15  25
Igrac B na potezu:
3*(5+25)
 = 90
Igrac A na potezu:
15*25+9*5
 = 420
Igrac A je osvojio rundu.

 Racunar: (5/3+25)*15-3 = 397

 -----> Ako zelite zavrsiti igru unesite Q,ostalo za nastavak:
```

примјер

Списак класа, изузетака и слободних функција

Класе:

- Генератор израза
- Израз
- Шаблон класа Калкулатор

Слободне функције (utils.cpp):

- void readFile(const std::string& filename, std::vector<std::vector<int>>& retVal)
- int evaluateExpression(const std::string& expr)
- int priority(char alpha)
- bool isValidExpression(const std::string& expr)
- void playGame()
- std::string infix_to_postfix(const std::string& infix, std::vector<int> numbers)
- void printGivenNumbers(const std::vector<int>& numbers, const int& target)

Инстанца класе **Израз** представља један од израза које генератор израза генерише. Битнији атрибути су **std::string expression** (запис израза), **double value** (вриједност израза), **bool valid** (валидност израза), **int num** (број операнада у изразу), **int priority** (приоритет операције), **std::vector<int> id** (искоришћени операнди) и **char sign** (знак операције).

Генератор израза има улогу да генерише све могуће изразе од понуђених бројева. У случају да нађе израз чија је вриједност једнака траженој, генерисање израза се прекида и враћа се дати израз као коначан резултат генерисања. У супротном, враћа израз чија је вриједност најближа траженом броју. Битнији атрибути су **int target** (тражена вриједност), **bool found** (да ли је пронађена), **int minValue** (минимална удаљеност од траженог броја), **Expression bestExp** (најбољи израз), **std::vector<int> operands** (вектор операнада) и вектори израза.

readFile(...) - чита податке из фајла који је прослеђен у командној линији

evaluateExpression(…) - рачуна вриједност израза

IsValidExpression(…) - провјерава да ли је унос валидан

PlayGame(…) - функција у којој се врши унос одговора такмичара, креира генератор и рјешење

infix_to_postfix(…) - израз који уноси такмичар пребацује у постфиксну нотацију

PrintGivenNumbers(…) - испис на почетку рунде тражени и понуђени број

СТРУКТУРА

Структура аргумената командне линије се састоји од једног стринга који представља назив фајла из ког се учитавају подаци за сваку рунду. (у нашем случају аргумент је input.txt)

Структура улазног фајла се састоји од редова у ком се налази 7 бројева одвојених “ “ сепаратором, гдје првих 6 бројева представља понуђене бројеве, а последњи број је тражени број.

```
1 5 8 9 10 25 456
3 5 9 3 15 25 397
4 7 7 4 20 75 592
9 8 6 8 15 25 712
3 9 2 2 10 75 244
8 7 5 1 15 25 496
2 2 5 9 20 50 503
9 9 7 8 15 100 901
```

примјер

Структура излазног фајла је таква да буде лако читљива кориснику.

```
//////////////////////////////////////////////////
Runda 1
Ponudjeni brojevi: 1 5 8 9 10 25
Trazeni broj: 456
Igrac A: (25-1)*10
Vrednost izraza: 240
Udaljenost: 216

Igrac B: (25-1)*(10+9)
Vrednost izraza: 456
Udaljenost: 0

Igrac B je nasao tacan broj. Osvojio je rundu.
Racunar: (25-1)*(9+10)
//////////////////////////////////////////////////
```

примјер исписа рунде у фајл

```
//////////////////////////KRAJ IGRE////////////////////////////////
Igrac A: 2 runde
Igrac B: 1 runde
Igrac A je pobijedio.
```

Примјер исписа побједника на крају игре

Опис алгоритма за тражење тачног решења

Алгоритам за проналажење тачног рјешења се заснива на рачунању вриједности најпростијих израза (садрже само један број) и генерисањем нових израза њиховим спајањем уз помоћ неког оператора (+,*,-,/). Затим се уз помоћ свих до тад генерисаних израза генеришу нови на сличан начин. Тај процес се понавља или док се не пронађе израз који даје тачно рјешење или док се не генеришу сви изрази који садрже свих 6 понуђених бројева па се врати онај чија је вриједност најближа траженој.

void ExprGenerator::generateExpr() има задатак да уз помоћ функција **buildOpComb(**) и **buildComb**(...) креира све могуће изразе које смјешта у један од 6 вектора **expN** (гдје N означава број од 1-6 и он представља колико је операнада искориштено за тај израз).

buildComb(...) има задатак да од израза у векторима **e1** и **e2** који су прослеђени направи нове које ће смјестити у прослеђени **e** вектор. Од два израза прави нови израз спајајући их помоћу одговарајућег оператора (+,*,-,/).

Ако је вриједност једног од два израза 1 онда се не спаја помоћу множења нити дијељења јер нема смисла. Такође се због комутативност врши спајање А+Б,А*Б али не и Б+А,Б*А. Тако се смањује број истих случајева.

bool ExprGenerator::newSolution(…) има задатак да провјери да ли је прослеђени израз ново најбоље рјешење. Ако је тачно рјешење онда прекида процес генерисања израза.

ОПИС НАЧИНА ТЕСТИРАЊА

За тестирање калкулатора израза и калкулатора генератора кориштене су функције **void TestCalculator()** и

void TestGenerator(). Тестирање је извршено тако што се у вектор унесу изрази и онда итеративно се провјерава резултат који даје функција за рачунање израза. Слично је тестирање и за калкукатор.

ПРОБЛЕМИ И ОГРАНИЧЕЊА

Први начин на који сам покушао да генеришем све изразе је рекурзивном функцијом, међутим било је превише случајева који се понављају па је приликом тражења одговарајућег рјешења долазило до heap overflowa. Затим сам имплементирао наведене класе, генератор израза и израз за проналажење рјешења.