Електротехнички факултет у Београду
Катедра за рачунарску технику и информатику

Предмет: Објектно оријентисано програмирање (1)
Школска година: 2024/2025. (Задатак важи почев од јануарског рока 2025.)

Пројектни задатак за домаћи рад

Верзија документа: 1.1

Садржај

Увод

Задатак овог пројекта јесте реализација једног релативно једноставног *интерпретера командне линије* (енгл. *command line interpreter*, CLI) који, иако значајно поједностављен у односу на оне који постоје у реалним оперативним системима као њихов интерфејс (енгл. *command line interface*, CLI), садржи мноштво функционалности и концепата које подржавају и реални оперативни системи. Овај интерпретер има задатак да учитава једну по једну команду са свог знаковног улаза и да сваку задату команду извршава. Команде корисник задаје интерактивно, али се оне могу задати и као *пакет* (енгл. *batch*) записан у неком текстуалном фајлу. Интерпретер подржава и концепте *редирекције* (енгл. *redirection*) и *цевовода* (енгл. *pipe*) уобичајене у оперативним системима. Сви наведени концепти и захтеви детаљно су описани у овом документу. Интерпретер треба реализовати на програмском језику C++.

Овај документ садржи пројектне захтеве за реализацију поменутог интерпретера. Уколико у овом документу нешто није прецизно дефинисано, студент треба да уведе разумну претпоставку, да је јасно и прецизно документује и да своје решење базира на њој.

Функционални захтеви

Основно понашање интерпретера

Интерпретер командне линије треба да се извршава као интерактивни програм који са корисником интерагује преко тзв. *конзоле* на датом оперативном систему – домаћину: корисник уноси знакове преко тастатуре, а програм свој одзив исписује као текст који се састоји од низа знакова редом исписаних у једном или више редова, уз преломе редова знаком за нови ред (`'\n'`). Овај програм улазне знакове треба да учитава са свог стандардног улаза, а да ипис даје на свој стандардни излаз.

Основна функционалност интерпретера испољава се као циклично понављање следећег поступка:

1. На почетку новог реда свог исписа, интерпретер исписује знак спермности (енгл. *command prompt*) за унос једне *командне линије* (енгл. *command line*); иницијално је то знак `'$'`, али се то може променити.

   ```
   $
   ```

2. Корисник уноси садржај командне линије као низ знакова који се завршава знаком за нови ред (`'\n'`). У основном облику, командна линија садржи једну команду. Команда може имати различите облике и прецизна синтакса и скуп команди биће дат касније, али је општи облик следећи:

   command `[-`*opt*`]` `[`*argument*`]`[1],

   где је *command* назив команде, *opt* опција која може утицати на модалитет извршења команде и која зависи од саме команде, а *argument* аргумент команде чије значење такође зависи од саме команде. На пример, следећа команда пребројава речи (`-w` за *word*) у тексту задатом аргументом под наводницима:

   ```
   $ wc -w "Lorem ipsum dolor sit amet, consectetur adipiscing elit"
   ```

3. Интерпретер извршава задату команду[2] која, у зависности од значења, евентуално исписује свој резултат на излаз као низ знакова, потенцијално у више редова:

   ```
   $ wc -w "Lorem ipsum dolor sit amet, consectetur adipiscing elit"
   8
   ```

4. По завршетку извршавања команди из командне линије, цео поступак се понавља – интерпретер поново исписује знак спремности за унос нове командне линије.

Лексичка правила

Запис једне командне линије подлеже следећим лексичким правилима:

1. Командна линија се састоји од низа знакова максималне дужине 512 знакова типа `char` који се завршава знаком за нови ред (`'\n'`); уколико се на улазу не наиђе на знак за нови ред до наведене дужине 512 знакова (која не укључује и знак за нови ред), сви

[1] Угласте заграде означавају необавезне делове записа и нису део самог записа команде.

[2] Или више команди у општијем случају, уколико је више команди задато у истој командној линији, како ће бити објашњено касније.

преостали знакови се игноришу, а као запис командне линије узима се унетих 512 знакова (не укључујући знак за нови ред).
2. Велика и мала слова се разликују.
3. Назив команде, њена опција и аргумент међусобно су раздвојени „белинама“ – секвенцама суседних знакова од којих је сваки од њих знак за размак (`' '`) или хоризонтални табулатор (`'\t'`).
4. Аргумент који је уоквирен наводницима (`"`) третира се као интегралан низ знакова који се прослеђује команди без растакања на речи раздвојене белинама: било који знак осим знака за нови ред (`'\n'`) и наводника (`"`) нема никакво посебно значење унутар наводника.
5. Знакови „усправна црта“ (`'|'`), „мање“ (`'<'`) и „веће“ (`'>'`) имају посебно значење које ће бити описано касније, осим ако се налазе унутар низа знакова уоквирених у наводнике, када се тумаче као саставни део тог низа знакова без посебног значења за интерпретер.

Ако у командној линији интерпретер наиђе на знак или више знакова који нису дозвољени на месту на ком се налазе, према наведеним лексичким правилима и синтакси команди описаних у овом документу, интерпретер је неће извршити и цела командна линија сматра се погрешном и не покреће извршавање ниједне команде из командне линије. У таквом случају интерпретер треба да испише поруку о грешци у којој треба да прикаже барем информацију о позицији првог неисправног знака на који је наишао. Оставља се имплементацији да поруку о грешци учини и информативнијом, рецимо тако што ће исписати најпре поруку о грешци, потом целу командну линију и испод свих погрешних знакова неки симбол (нпр. `^`) који помаже кориснику да погрешне знакове уочи. На пример:

```
$ wc& -w *"Lorem ipsum dolor sit amet" +?

Error - unexpected characters:
wc& -w *"Lorem ipsum dolor sit amet" +?
  ^    ^                          ^^
```

Улазни и излазни ток команде

Већина команди (али не све) ради над секвенцама знакова које узимају са свог *улазног знаковног тока* (енгл. *input character stream*), који представља уопштење следећих случајева:

1. Подразумевано, ако команда нема свој аргумент, улазни ток команде је конзола, тј. тастатура: команда узима улазне знакове које корисник откуца на тастатури током извршавања команде. Унос знака који стандардна библиотека представља симболичком константом `EOF` (истовремени притисак тастера Ctrl и Z на Windows системима, односно тастера Ctrl и D на системима налик систему Unix) завршава улазну секвенцу знакова, односно завршава улазни ток. На пример, након што се покрене ова команда:

   ```
   $ wc -w
   ```

   команда очекује да корисник унесе низ знакова на тастатури. Ако корисник откуца
   ```
   Lorem ipsum dolor sit amet
   ```
   и након тога притисне комбинацију тастера за завршетак улазног тока[3]:
   команда ће на екрану исписати само број 5.

[3] Ови знакови се не виде на екрану, јер то није дефинисано понашање ове команде, пошто она на свој излазни ток не исписује знакове са свог улазног тока, већ само резултат свог рада.

2. Ако команда има свој аргумент који је уоквирен у наводнике (`"`), целокупни низ знакова између наводника (без тих наводника) је улазни ток команде. На пример, након што се покрене ова команда:

   ```
   $ wc -w "Lorem ipsum dolor sit amet"
   ```

   команда ће на екрану исписати број 5.
3. Ако команда има свој аргумент који није уоквирен у наводнике, тај аргумент представља име текстуалног фајла (заправо стазу до тог фајла у односу на текући директоријум у ком се извршава интерпретер); садржај тог фајла, посматран као текстуални запис, тада представља улазни ток команде. На пример, након што се покрене ова команда:

   ```
   $ wc -w input.txt
   ```

   и ако је садржај фајла `input.txt` следећи:
   ```
   Lorem ipsum dolor sit amet
   ```
   команда ће на екрану исписати само број 5.
4. Само ако команда нема аргумент који јој дефинише улазни ток као у претходном случају, њен улазни ток може се *преусмерити* (енгл. *redirect*) на текстуални фајл, тако да команда садржај тог фајла узима као свој улазни ток на исти начин као у претходном случају. Ознака за редирекцију улазног тока је знак `'<'`. На пример, након што се покрене ова команда:

   ```
   $ wc -w <input.txt
   ```

   и ако је садржај фајла `input.txt` следећи:
   ```
   Lorem ipsum dolor sit amet
   ```
   команда ће на екрану исписати само број 5.

Аналогно улазном знаковном току, команда има и свој *излазни знаковни ток* (енгл. *ouptut character stream*) – место на које исписује знакове које даје као свој излаз тј. резултат и које може бити следеће:

1. Подразумевано је излазни ток конзола, тј. екран: команда исписује свој излазни резултат на екран и интерпретер након тога пребацује курзор у нови ред за прихват следеће командне линије. На пример, након што се покрене ова команда:

   ```
   $ wc -w "Lorem ipsum dolor sit amet"
   ```

   команда ће на екрану исписати број 5.
2. Излазни ток команде може се *преусмерити* (енгл. *redirect*) на текстуални фајл, тако да команда свој излаз уписује у садржај тог фајла. Ако дати фајл не постоји, команда ће га направити. Ако тај фајл већ постоји, команда ће његов претходни садржај потпуно обрисати и у њега изнова исписати свој резултат. Ознака за редирекцију излазног тока је знак `'>'`. На пример, након што се покрене ова команда:

   ```
   $ wc -w "Lorem ipsum dolor sit amet" >output.txt
   ```

   садржај фајла `output.txt` ће бити само знак 5.
   Уколико се као знак за редирекцију излазног тока појави двоструки знак `'>'` без иједног знака између, излаз команде ће бити додат на крај садржаја излазног фајла, а претходно постојећи садржај тог фајла неће бити обрисан уколико постоји.

Део за редирекцију може се наћи само на крају целе команде. Команди која има аргумент који јој дефинише улазни ток не може се преусмерити улазни ток. Испред и иза знакова за редирекцију (`'<'`, једноструки `'>'` и двоструки `'>'`) може, али не мора бити један или више знакова који се по наведеним лексичким правилима сматрају белинама. Под

условима наведеним за улазну и излазну редирекцију, команди се може преусмерити и улазни и излазни ток; редослед редирекција иза команде је произвољан (дозвољавају се оба).

Назив фајла представља низ суседних знакова који се завршава знаком који се по лексичким правилима сматра белином. Овај назив интерпретира оперативни систем домаћин у операцији отварања фајла, па дати низ знакова треба проследити систему у операцији отварања фајла без измена.

Команде

У наставку је дат списак команди које треба подржати. Систем је предвиђен за будућа проширења новим командама, и то како оних које имају исти општи наведени формат са опцијом и аргументом, али и општији формат, са произвољним додацима иза самог назива команде којим командна линија увек почиње. Такође се могу уводити нове опције за постојеће команде.

Команда `echo`

Формат:

`echo [argument]`

Опис: Прослеђује знакове са свог улазног тока на свој излазни ток без икаквих измена.

Опције: Нема.

Команда `prompt`

Формат:

`prompt argument`

Опис: Мења (поставља) знак за спремност интерпретера за учитавање нове командне линије (енгл. *command prompt*) на низ знакова који је задат аргументом уоквиреним у наводнике.

Опције: Нема.

Команда `time`

Формат:

`time`

Опис: На свој излазни знаковни ток исписује текуће време на системском часовнику реалног времена.

Опције: Нема.

Команда `date`

Формат:

`date`

Опис: На свој излазни знаковни ток исписује текући датум на системском часовнику реалног времена.

Опције: Нема.

Команда `touch`

Формат:

`touch` *`filename`*

Опис: Прави фајл са задатим именом и празним садржајем у текућем директоријуму (директоријум у ком се извршава интерпретер). Уколико тај фајл већ постоји, исписује поруку о грешци и нема никакав други ефекат.

Опције: Нема.

Команда `truncate`

Формат:

`truncate` *`filename`*

Опис: Брише садржај фајла са задатим именом из текућег директоријума.

Опције: Нема.

Команда `rm`

Формат:

`rm` *`filename`*

Опис: Брише фајл са задатим именом (уклања фајл из фајл система) из текућег директоријума.

Опције: Нема.

Команда `wc`

Формат:

`wc -`*`opt`* `[`*`argument`*`]`

Опис: Броји речи или све знакове у тексту учитаном са улазног знаковног тока и број исписује на свој излазни ток. Речима се сматрају секвенце суседних знакова раздвојене белинама, при чему се белином за ову команду сматра знак за који стадардна библиотечна функција `std::isspace` враћа `true`.

Опције:

- `-w` Броји речи
- `-c` Броји све знакове

Команда `tr`

Формат:

`tr [`*`argument`*`] what [`*`with`*`]`

Опис: У тексту прочитаном са улазног тока проналази све појаве низа знакова *`what`* уоквиреног наводницима низом знакова *`with`* уоквиреног наводницима и резултујући текст исписује на свој излазни ток. Уколико низ знакова *`with`* није задат, појаве низа *`what`* се просто уклањају из улазног текста.

Опције: Нема.

Команда `head`

Формат:

`head -n`*`count`* `[`*`argument`*`]`

Опис: Из текста прочитаног са улазног тока преноси првих неколико линија на свој излазни ток, а преостали садржај улазног тока игнорише.

Опције: Обавезна опција `-n` задаје број првих линија текста који се преноси са улаза на излаз. Иза знакова `-n` одмах (без икаквих знакова између) следи низ од највише 5 децималних цифара које одређују број редова.

Команда `batch`

Формат:

`batch [`*`argument`*`]`

Опис: Садржај улазног тока интерпретира као произвољно дуг низ командних линија раздвојених знаком за нови ред и интерпретира све те командне линије као што је описано у овом документу, исто као да су оне учитане са конзоле (тзв. *пакетна обрада*, енгл. *batch*). Интерпретира независно једну по једну командну линију све док не наиђе на крај улазног тока. Уколико је нека командна линија изазвала грешку, исписује њену поруку о грешци на свој излазни ток и наставља даље на следећу. Улазни ток може садржати било које команде, па и саму команду `batch`; рекурзију не треба спречавати, последице њеног извшравања су одговорност корисника.

Опције: Нема.

Цевоводи

Једна командна линија може садржати више команди повезаних у тзв. *цевовод* (енгл. *pipe*): излазни ток једне команде повезује се као улазни ток наредне команде наведене у командној линији; тиме сваки знак који прва команда избаци на свој излазни ток бива прослеђен другој команди на њен улазни ток, чувајући поредак знакова. Свака команда наведена је у свом дефинисаном формату, а ознака за повезивање у цевовод је усправна црта (`'|'`).

На пример, следећа командна линија садржи три команде повезане у цевовод:

```
time | tr ":" "." | wc -c > time.txt
```

Излазни ток прве команде `time` везан је на улазни ток друге команде `tr`, тако да све знакове које ова команда избаци на свој излазни ток добија друга команда `tr` на свој улазни ток. Ова друга команда замењује знак `':'` знаком `'.'`, а тако трансформисан текст добија трећа команда `wc` на свој улаз. Она опет пребројава све знакове у том тексту и резултат уписује у фајл `text.txt`.

Команда чији је улазни или излазни ток повезан у цевовод не може имати свој аргумент који дефинише тај исти ток на другачији начин (као непосредан текст, фајл или редирекцијом). Знак за повезивање у цевовод може, али не мора бити окружен белинама.

Третман грешака

Све грешке на које се наиђе током лексичке и синтаксне анализе или извшавања команде треба да резултују исписом поруке о грешци на излазни ток и одустајањем од извршавања те команде без икаквог ефекта. Грешке обухватају следеће категорије:

1. Лексичке грешке обрађују се како је наведено раније.
2. Непостојећа команда. Ова грешка треба да испише поруку: "`Unknown command: command`", уз испис низа знакова који стоји на месту на ком се очекује команда која није препозната.
3. Синтаксне грешке у формату команде или целе командне линије.
4. Семантичке грешке у дефинисању улазног или излазног знаковног тока, нпр. команда која има и аргумент и редирекцију улазног тока или команда чији је улазни или излазни ток везан у цевовод и дефинисан на други начин.
5. Семантичка грешка у извршавању команде, ако је то дефинисано специфично за поједине команде.
6. Грешке које је пријавио оперативни систем током извршавања команде, нпр. при раду са фајловима (фајл не постоји, а очекује се да постоји, грешка у правима приступа и било која друга грешка).

Остали захтеви

Опсег фаза пројекта

У првој фази пројекта треба реализовати следећи подскуп захтева:

- Извршавање само команди `echo`, `time`, `date`, `touch` и `wc`.
- Подршку за аргументе и све опције наведених команди, укључујући улазни ток из непосредног аргумента уоквиреног наводницима или из текстуалног фајла.
- Обраду само грешака типова наведених под тачкама 5 и 6 у одељку ”Третман грешака”. Сви тест примери у овој фази биће лексички, синтаксно и семантички исправни, осим што евентуално могу узроковати грешке под тачкама 5 и 6.
- Не треба подржати редирекцију улазног и излазног тока нити цевовод.

У другој фази пројекта треба имплементирати све захтеве наведене у овом документу.

Начин провере

Да би био прегледан, пројекат мора да се преводи и повезује без иједне грешке. Пројекат који има грешку у превођењу или повезивању биће одбачен без даљег прегледања. Препоручује се решавање свих евентуалних упозорења преводиоца (енгл. *warning*).

Да би био прегледан, програм мора да изврши све јавне тестове без грешака. Програм који има грешку у јавним тестовима биће одбачен без даљег прегледања. Јавни тестови и њихови исправни резултати ће бити познати студентима при изради пројекта.

Након што је прошао јавне тестове, програм ће бити подвргнут тестирању помоћу тајних тестова (непознатих унапред). Уколико програм не прође неки од јавних тестова, пројекат може бити одбачен у целини или бодован одређеним бројем негативних поена, у зависности од врсте теста и озбиљности грешке.

Након спровођења тајних тестова, а уколико до тада пројекат није одбачен или оцењен са 0 поена, пројекат ће бити прегледан од стране испитивача који ће оцењивати његов квалитет. Аспекти који ће бити укључени у преглед и оцену су следећи:

1. Стил и уредност програма:
   - Назубљивање (увлачење) записа
   - Једнобразност стила
   - Јасноћа и једнобразност именовања (идентификатори, велика и мала слова)
   - Јасно раздвајање логичких целина програмског записа (прореди на одговарајућим местима који одвајају логичке целине у дефиницијама класа, телима потпрограма и у фајлу)
   - Коментари на одговарајућим местима (ни превише, ни недовољно).

2. Објектна декомпозиција:
   - Апстракција: јасни и добро дефинисани концепти (апстракције, класе)
   - Расподела одговорности по класама: добро и јасно дефинисане одговорности класа и распоређене према правилима интерне кохезије и екстерне слабе повезаности
   - Модуларност: декомпозиција на модуле (фајлове), јасно и добро дефинисани интерфејси модула (заглавља и њихов садржај)

- Енкапсулација на нивоу класа: јасно и добро дефинисани интерфејси и имплементације класа
- Хијерархијска декомпозиција: добро дефинисане хијерархије класа, разумно коришћење наслеђивања
- Механизми сарадње: добро дефинисани и лако разумљиви механизми интеракције, лако праћење сценарија сарадње
- Релације између класа: јасне зависности између класа и везе између објеката
- Алгоритамска декомпозиција: подела на потпрограме на добар начин, кратка тела потпрограма
- Свеукупна разумљивост пројекта и лакоћа одржавања (измене и проширења).

3. Техника објектно оријентисаног програмирања и владање језиком C++

Ова категорија оцењује исправно коришћење концепата и техника ООП и језика C++ које су покривене градивом курса. Треба имати у виду да ово *никако не значи* да програм треба да укључи *све* те концепте и технике, напротив: употреба неког концепта или технике тамо где није примерено или потребно може бити оцењена као лоша. Студент не треба да по сваку цену употребљава све што је научио, како би то показао, већ да технике и концепте употребљава тамо где је то потребно и оправдано, а да за разлог има добро образложење. Концепти и технике чија се употреба оцењује укључује све оно што је покривено наставом, на пример (списак није комплетан):

- класе, чланови класа
- извођење класа (једноструко/вишеструко, јавно/заштићено/приватно)
- преклапање оператора
- показивачи и референце
- константност
- конструктори
- управљање објектима који имају алоциране ресурсе (конструктори копирања/премештања, оператори доделе копирањем/премештањем, деструктор)
- управљање алокацијом меморије
- обрада изузетака
- перформансе

и слично.

Записник ревизија

Верзија 1.1

Стране	Измене
Све	Документ написан
8.	Командама `touch`, `rm` и `truncate` промењен назив аргумента у `filename` и додато појашњење.