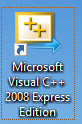
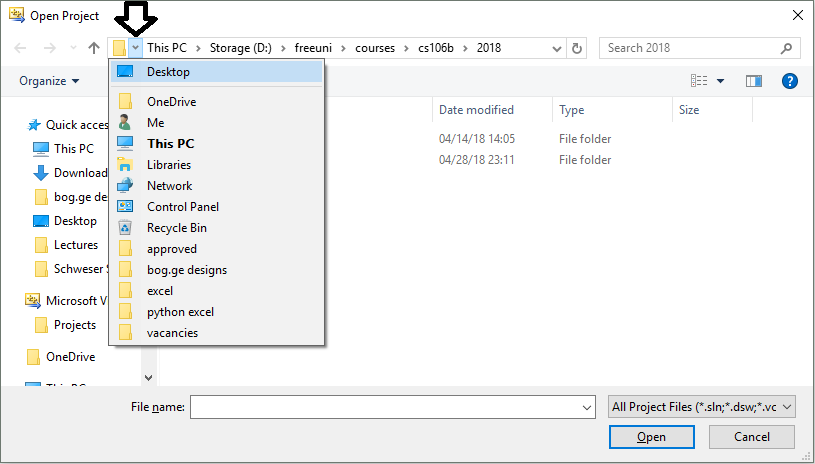
|  |  |
| --- | --- |
| საგანი | პროგრემირების აბსტრქციები |
| პროგრამა | MACS |
| კურსი | I, II |
| ჯგუფი/ჯგუფები |  |
| ლექტორი | გიორგი ბოჭორიშვილი |
| **გამოცდის ფორმატი (მონიშნეთ)** | |
| ☐ დახურული წიგნი | |
| ღია წიგნის შემთხვევაში მონიშნეთ | ☒ ელექტრონული მასალა  ☒ წიგნები |
| გამოცდის კომპიუტერში წერის შემთხვევაში,მონიშნეთ მხოლოდ ის კონკრეტული პროგრამული უზრუნველყოფა (ერთი ან რამდენიმე),რომელიც უნდა გამოიყენოს სტუდენტმა.  ***ნაშრომების შეგროვება და მოწოდება მოხდება , თქვენ მიერ მითითებული პროგრამის გაფართოების მიხედვით.*** | ☒ **Word** (\*.docx; \*.doc)  ☐ **Excel** (\*.xlsx; \*.xls)  ☐ **PowerPoint** (\*.pptx; \*.ppt)  ☐ **Oracle Java** (\*.java)  ☐ **IBM SPSS Statistics 20** (\*.sav)  ☐ **SPSS Statistics 17.0** (\*.spv; \*.sav)  ☐ **ISIS 7 Professional** (\*.DSN)  ☐ **Multisim 12.0** (\*.ms12)  ☐ **ARES 7 Professional** (\*.LYT)  ☐ **Flowcode V4 for AVRs** (\*.fcf\_avr)  ☐ **logisim-generic-2.7.1** (\*.circ)  ☐ **GeoGebra** (\*.ggb)  ☐ **Matlab 2016a** (\*.m; \*.mat)  ☐ **EpiInfo** (\*.prj)  ☐ **LCD Vision**  ☒ **Microsoft Visual C++ 2008 Express Edition** (\*.vcproj)  ☐ **Codevision AVR** (\*.c)  ☐ **PHStat2** (\*.xlsx)  ☐ **Kawa** (\*.\*)  ☒ **Dev-C++** (\*.cpp;\*.cc;\*.cxx;\*.c++;\*.cp; \*.c)  ☒ **Visual Studio Code** (\*.cpp; \*.c; \*.java; \*.sql; \*.jsx; \*.js)  ☒ **Visual Studio 2015**  ☐ **Gcc** (\*.c; \*.cpp; \*.java)  ☐ **Eclipse IDE 9** (\*.txt; \*.cpp; \*.c; \*.java; \*.sql; \*.jsx; \*.js)  ☒ **Notepad ++** (\*.txt; \*.cpp; \*.c; \*.java; \*.sql; \*.jsx; \*.js)  ☐ **სხვა \_** (გაფართოება) |
| თუ სტუდენტი ერთდროულად მუშაობს რამოდენიმე პროგრამაში და საბოლოო სახით, საჭიროა ნაშრომების დაარქივება, გთხოვთ მონიშნოთ შესაბამისი გრაფა. | ☒\*.zip;  ☒\*.rar; |
| ფურცელზე წერის შემთხვევაში მონიშნეთ შესაბამისი | ☐ სრულად ფურცელზე  ☐ ფურცელზე და კომპიუტერში |

**\*მონიშნეთ გამოცდის ფორმატი (მიუთითეთ √)**

შუალედური გამოცდა

პროგრამირების აბსტრაქციებში

**შეასრულეთ შემდეგი ინსტრუქციები, წინააღმდეგ შემთხვევაში შესაძლოა თქვენი ნაშრომი არ შეფასდეს.**

1. **ჩამოტვურთტეთ საგამოცდო საკითხები და თქვენი ვარიანტის შესაბამისი პროექტის არქივი თქვენს დესკტოპზე.**
2. **ამოაარქივეთ პროექტი. მაუსის მენიუ ღილაკით დააჭირეთ არქივზე და აირჩიეთ Extract here.**
3. **დესკტოპზე იპოვეთ Microsoft Visual C++ 2008 ის გამშვები ფაილი და გაუშვით.  
   **
4. **File მენიუდან აირჩიეთ Open > Project/Solution, გაიხსნება ფანჯარა სადაც მარცხნივ ჩამონათვალში აირჩიეთ Desktop. თუ ჩამონათვალში არ არის Desktop დააჭირეთ სურათზე აღნიშნულ ისარს და აირჩიეთ.   
   **
5. **შემდეგ აირჩიეთ ამოარქივებული ფოლდერში project\Blank Project.sln ფაილი და დააჭირეთ Open ს.**
6. **თუ Visual Studio ში არ გიჩანთ ფაილები, View მენიუში მონიშნეთ Solution Explorer პუნქტი**
7. **თითოეული ამოცანისთვის დაგხვდებათ შესაბამისი \*.cpp ფაილი, სადაც უნდა დაწეროთ თქვენი ამოხსნა. ფუნქციის პროტოტიპის შეცლა არ შეიძლება. შეგიძლიათ შემოიტანოთ დამხმარე ფუნქციები. არ გაქვთ გლობალური ცვლადების გამოყენების უფება.**
8. **main.cpp ფაილში წერია ამოცანების ტესტებზე შემოწმება. მისი გაშვებით ხდება ამოცანებში მოცემული ფუნქციების გამოძახება სხვადასხვა input-ით და დაბრუნებული პასუხის შემოწმება. კონსოლში გამოვა ინფორმაცია თუ რომელ ტესტებზე გაიარა\ჩაიჭრა თქვენმა ამოხსნამ.**
9. **ამოხსნის დასრულების შემდეგ დაარქივეთ „problems” ფოლდერი და არქივს დაარქვით სახელად თქვენი მეილის პრეფიქსი, მაგალითად gboch10.zip**
10. **არქივი დატოვეთ დესკტოპზე თქვენს ფოლდერში**

**საყურადღებო:**

1. კოდი გაუშვით f5 ღილაკზე დაჭერით ან მწვანე Play ღილაკზე დაჭერით, წინააღმდეგ შემთხვევაში კონსოლის ფორმა არ გამოჩნდება.
2. Fatal error LNK1168 cannot open - ამ შეცდომის შემთხვევაში დამიძახეთ მე, ან საგამოცდოს რომელიმე თანამშრომელს. თქვნს კომპოიყერზე საჭირო იქნება Blank Project პროცესის მოკვლა რაშიც IT ის თანამშრომელი დაგეხმარებათ.
3. თუ კოდში ცვლილება შეგაქვთ მაგრამ გაშვებისას არანაირად არი ისახება, დარწმუნდით რომ წინა წარმატებული ბილდი არ ეშვება. F7 ით ცადეთ დაკომპილირება.
4. თუ გსურთ ხაზების ნუმერაციის გამოჩენა, გახსენით Tools > Options მენიუ, გადადით Text Editor > All Languages > General და მონიშნეთ Line numbers.
5. თუ გაურკვეველი შეცდომა გაქვთ, ცადეთ მენიუდან Build > Clean Solution და Build > Clean Blank Project

**ამოცანა 1. (120 ქულა) LinkedHashMap**

თქვენი ამოცანაა დაწეროთ LinkedHashMap-ის იმპლემენტაცია, რომელიც HashMap-ისგან მხოლოდ იმით განსხვავდება რომ ინახავს key – value წყვილების ჩამატების რიგითობას და იტერატორს შეუძლია ზუსტად იმ მიმდევრობით დაგვიბრუნოს key ები, რა მიმდევრობითაც მოხდა ჩამატება. hashCode ფუნქცია მოცემული გაქვთ.

put, containsKey, get, remove და size ფუნქციები ზუსტად იგივენაირად მუშაობენ როგორც სტანდარტულ HashMap-ის შემთხვევაში.

void resetKeyIterator() - ამ ფუნქციის გამოძახებს შემდეგ იტერატორი დგება სიის თავში

bool hasNextKey() - ამოწმებს დარჩენილია თუ არა სიაში Key

string nextKey() - აბრუნებებს Key-ებს ზუსტად იმ მიმდევრობით, რა მიმდევრობითაც მოხდა ჩამატება, ანუ პირველად დააბრუნებს Key-ს რომელიც ყველაზე ადრე დაემატა put ფუნქციის საშუალებით. შემდეგ დააბრუნებს Key-ს რომელიც მომდევნო დაემატა.

Key ებზე იტერირების პროცესის წარმოსადგენად ქვემოთ მოცემულია კოდი თუ როგორ უნდა დავბეჭდოთ მეფში არსებული Key – Valye წყვილები:

map.resetKeyIterator();

while (map.hasNextKey()) {

string key = map.nextKey();

cout << key << “-” << map.get(key) << endl;

}

იმისათვის რომ უფრო კარგად გაიგოთ რას ნიშნავს რიგითობის შენარჩუნება აქვე მოვიყვანე მაგალითი:

LinkedHashMap map;

map.put(“key1”, 1);

map.put(“key2”, 2);

map.put(“key3”, 3);

map.resetKeyIterator();

while (map.hasNextKey()) {

cout << map.nextKey() << “ “;

} //დაიბეჭდება 1 2 3

map.put(“key1”, 1);//თავზე გადაეწერება ძველ მნიშვნელობას ახალი და რიგითობაც შეიცვლება

map.resetKeyIterator();

while (map.hasNextKey()) {

cout << map.nextKey() << “ “;

} //დაიბეჭდება 2 3 1

map.remove(“key3”);//წაიშლება key3 და მისი შესაბამისი მნიშნელობა

map.resetKeyIterator();

while (map.hasNextKey()) {

cout << map.nextKey() << “ “;

} //დაიბეჭდება 2 1

დამატებითი მოთხოვნები:

* ყველა ფუნქცია უნდა მუშაობდეს O(1) დროში.
* მეფში საწყისი ბაქეთების რაოდენობა უნდა იყოს kNumBuckets
* ელემენტების რაოდენობების და ბაქეთების რაოდენობის შეფარდება არ უნდა იყოს 2 ზე მეტი.
* არ გაქვთ უფლება გამოიყენოთ რომელიმე მზა ქოლქშენ სტრუქტურა, მასივის გამოყენება შეგიძლიათ

ამოცანის გასამარტივებლად შემოგთავაზებთ იმპლემენტაციის ერთერთ ალგორითმს, თუმცა არ ხართ ვალდებული ამ მიდგომით დაწეროთ რეალიზაცია.

ჰეშმეფ სტრუქტურის იმეპლემენტაცია გარჩეული გვაქვს ასე რომ ამ მხრივ ახალს ვერაფერს გეტყვით. შედარებით რთული ნაწილი არის ელემენტების რიგითობის შენახვა და ამისთვის ორი რამ დაგჭირდებათ:

1. განსაზღვროთ ორმაგი ბმული სიის სტრუქტურა(DoublyCell), რომელშიც შეინახავთ Key-Value ებს ზუსტად იმ მიმდევრობთ რა რიგითობითაც მოხდება მათი მეფში ჩამატება.
2. ბაქეთების სტრუქტურაში(Cell), სადაც ინახავთ Key, Value მნიშვნელობებს ახალი ველის ჩამატება, რომელშიც შეინახავთ ორმაგ ბმულ სიაში იმ ელემენტის მისამართს რომელშიც ინახება ზუსტად იგივე Key.

struct DoublyCell {

string key;

int value;

DoublyCell \* next;

DoublyCell \* prev;

};

struct Cell {

string key;

int value;

Cell \* next;

DoublyCell \* place;

};

ახალი Key Value ს დამატების დროს, თუ Key არ არსებობს მეფში:

1. ქმნით და ამატებთ შესაბამის ბაქეთში ახალი Cell ელემენტს
2. ქმნით და ამატებთ ორმაგი ბმული სიის ბოლოში ახალი DoublyCell ელემენტს
3. ახლად შექმნილ Cell სტრუქტურის შესაბამის ველს მიანიჭებთ ახლად შექმნილ DoublyCell ის მისამართს

ახალი Key Value ს დამატების დროს, თუ Key უკვე არსებობს მეფში:

1. იპოვით Key ს შესაბამის Cell სტრუქტურას და განაახლებთ value ს
2. Cell სტრუქტურიდან მარტივად ამოიღებთ ორმაგ ბმულ სიაში შესაბამისი ელემენტის მისამართს და გადაიტანთ მას ორმაგი ბმული სიის ბოლოში

მეფიდან Key ს წაშლის დროს:

1. იპოვით Key ს შესაბამის Cell სტრუქტურას
2. ამოიღებთ შესაბამისი DoublyCell ელემენტის მისამართს და წაშლით მას ორმაგი სიიდან
3. წაშლით Key ს შესაბამის Cell სტრუქტურას

Main.cpp ფაილში შეგიძლიათ იხილოთ ტესტების მაგალითები, ბოლო 4 ტესტი გათვლილია ფუნქციების ეფექტურობის შემოწმებაზე. ჩათვალეთ რომ იტერირების პროცესში არ მოხდება put, remove ფუნქციების გამოძახება. ასევე არ მოხდება get ან nextKey ს გამოძახება როცა მეფში არ არის ელემენტი.