***გამოცდის ფორმატი***

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

|  |  |
| --- | --- |
| დახურული წიგნი |  |
| ღია წიგნი | **√** |
| **\*ღია წიგნის შემთხვევაში მონიშნეთ გამოცდაზე ნებადართული ელემენტები (მიუთითეთ √)** | |
| სალექციო მასალები (პრეზენტაცია და სხვა) |  |
| ელექტრონული წიგნები | **√** |
| წიგნები |  |
| კონსპექტები |  |
| ლექსიკონი |  |
| კალკულატორი |  |
| ლეპტოპი/პლანშეტი |  |

**\* გამოცდის ჩატარების წესი იხილეთ ,,დესკტოპზე“ საქაღალდეში Exam materials**

***საგამოცდო საკითხების ფორმა***

***ვარიანტი # 1***

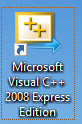
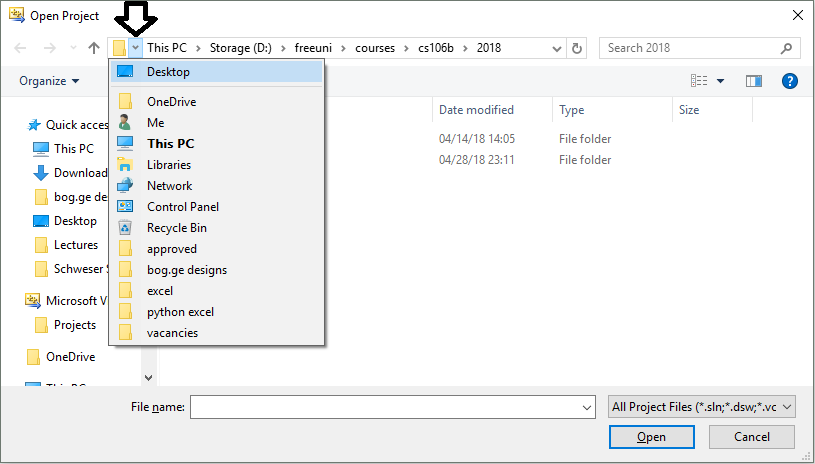
|  |  |  |  |
| --- | --- | --- | --- |
| ***სკოლა*/*საგანმანათლებლო პროგრამა*** | მათემატიკა და კომპიუტერული მეცნიერება | ***სტუდენტის მიერ მიღებული ქულა*** |  |
| ***საგანი*** | პროგრამირების აბსტრაქციები | | |
| ***ლექტორი*** | გ. ბოჭორიშვილი | | |
| ***კურსი*** | I, II | | |
| ***ჯგუფი*** |  | | |
| ***გამოცდის ფორმა*** | ღია წიგნი | | |
| ***გამოცდის***  ***ხანგრძლივობა*** | 2.5 საათი | | |
| ***მაქსიმალური ქულა*** | 180 | | |
| ***სტუდენტის სახელი და გვარი:*** | | | |

სახელი: ქულა:

ფინალური გამოცდა

პროგრამირების აბსტრაქციებში

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

1. **ჩამოტვირთეთ საგამოცდო საკითხები და თქვენი ვარიანტის შესაბამისი პროექტის არქივი midterm1.zip 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 დააჭირეთ სურათზე აღნიშნულ ისარს და აირჩიეთ Desktop,   
   **
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. main.cpp ფაილში წერია ამოცანების ტესტებზე შემოწმება და მისი გაშვებით კონსოლში გამოვა ინფორმაცია თუ რომელ ტესტებზე გაიარა\ჩაიჭრა თქვენმა ამოხსნამ. ფუნქციის პროტოტიპის შეცლა არ შეიძლება. შეგიძლიათ შემოიტანოთ დამხმარე ფუნქციები. არ გაქვთ გლობალური ცვლადების გამოყენების უფება.**
8. **ამოხსნის დასრულების შემდეგ დაარქივეთ „Desktop\midterm 2\problems” ფოლდერი და არქივს დაარქვით სახელად თქვენი მეილის პრეფიქსი, მაგალითად gboch10.zip**
9. **ვებ ბრაუზერში გახსენით მისამართი** [http://192.168.210.5](http://192.168.210.5/) **და ატვირთეთ არქივი.**

**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 ctrl + shift + esc ღილაკებით გახსენით task manager და მოკალით ყველა Blank Project პროცესი
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.

**ამოცანა 1. მეფი რიგითობის შენარჩუნებით (120 ქულა)**

თქვენი ამოცანაა დაწეროთ OrderedMap ის რეალიზაცია. სტრუქტურა თქვენთვის ნაცნობი მეფისგან ბევრით არაფრით განსხვავდება. ქვემოთ იხილეთ სტრუქტურის ინტერფესი და მეთოდების განმარტება.

**OrderedMap(); - O(1)**

კონსტრუქტორი, ქმნის ცარიელ მეფს

**void put(int key, int value); - O(1)**

მეფში key – value წყვილის დამატება, თუ ასეთი გასაღებით წყვილი უკვე არსებობს, ხდება მნიშვნელობის განახლება.

**int get(int key); - O(1)**

გასაღების მიხედვით აბრუნებს მნიშვნელბას, ჩათვალეთ ყოველთვის ისეთი გასაღები გადაეცემა რომელიც უკვე ინახება მეფში.

**int size(); - O(1)**

აბრუნებს მეფში შენახული წყვილების რაოდენობას.

**void remove(int key); - O(1)**

შლის მეფიდან გადაცემული გასაღების შესაბამის წყვილს, თუ ასეთი არ არსებობს არაფერი არ ხდება.

**bool containsKey(int key); - O(1)**

აბრუნებს true ს თუ გადაცემული გასაღების შესაბამისი წყვილი არსებობს მეფში, თუ არადა false ს.

მომდევნო სამი მეთოდი ასრულებს სტანდარტული მეფის იტერატორის ფუნქციონალს, ერთადერთი იმ განსხვავებით რომ OrderedMap ში იტერირება ხდება იმის მიხედვით თუ რა რიგითობით მოხდა მეფში ელემენტების ჩამატება. უფრო კონკრეტულად კი, პირველი ბრუნდება ის ელემეტი რომელიც ბოლოს დაემატა, მეორე რომელიც ბოლოს წინა და ასე შემდეგ, ბოლოს ბრუნდება ის ელემტი რომელიც სულ პირველად ჩაემატა მეფში.

**void resetIterator(); - O(1)**

იტერაციას არესეტებს, ანუ იწყებს ისევ თავიდან.

**bool hasNext(); - O(1)**

აბრუნებს true ს თუ კიდევ დარჩა დასაბრუნებელი ელემენტი მეფში, წინააღმდეგ შემთხვევაში false ს.

**int next(); - O(1)**

აბრუნებს რიგით მომდევნო ელემენტს.

ბოლო სამი მეთოდი შინაარსის უკეთესად გასააზრებლად იხილეთ მაგალი:

OrderedMap map;

map.put(1, 101);

map.put(2, 102);

map.put(1, 111);

map.put(3, 103);

map.put(4, 104);

cout << map.next() << endl;// 4

cout << map.next() << endl;// 3

cout << map.next() << endl;// 2

cout << map.next() << endl;// 1

cout << map.hasNext() << endl;// false

map.resetIterator();

cout << map.next() << endl;// 4

cout << map.next() << endl;// 3

map.resetIterator();

cout << map.next() << endl;// 4

ჰეშკოდათ შეგიძლიათ პირდაპირ გასაღებად გადაცემული რიცხვითი მნიშვნელობა აიღოთ, სხვა ჰეშირების ფუნქციის მოფიქრება არ გევალებათ.

ამოცანა შემოწმდება რამდენიმე ნაწილად, ქვემოთ მოცემულია მათი წონები:

1. put, get, size და containsKey მეთოდები, წონა 40%
2. remove მეთოდი, წონა იქნება 20%
3. next, hasNext, resetIterator, წონა 40%

გარდა ამისა თუ რომელიმე მეთოდი არღვევს დროითი სირთულის შეზღუდვას, შესაბამის კომპონენტში დაგაკლდებათ 60%. ანუ მაგალითად remove-ის ტესტირების დროს თუ put მეთოდი O(log n) ზე უფრო დიდ დროს ანდომებს, 20% ის ნაცვლად მიიღებთ 8% ს.

მზა Collection ტიპის სტრუქტურების გამოყენების უფლება არ გაქვთ, შეგიძლიათ გამოიყენოთ მასივი, ან ნოლიდან დაწეროთ თქვენთვის საჭირო სტრუქტურა და შემდეგ გამოიყენოთ. ამ პირობის დარღვევის შემთხვევაში ამოცანა განულდება.