***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***

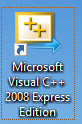
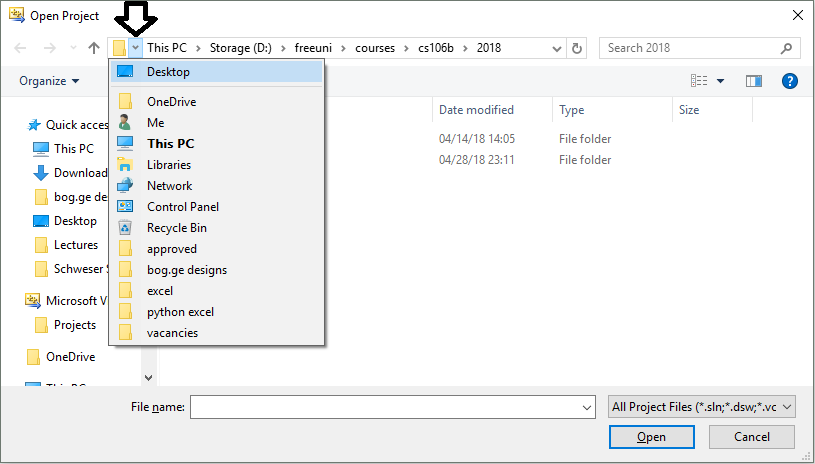
|  |  |  |  |
| --- | --- | --- | --- |
| ***სკოლა*/*საგანმანათლებლო პროგრამა*** | მათემატიკა და კომპიუტერული მეცნიერება | ***სტუდენტის მიერ მიღებული ქულა*** |  |
| ***საგანი*** | პროგრამირების აბსტრაქციები | | |
| ***ლექტორი*** | გ. ბოჭორიშვილი | | |
| ***კურსი*** | I, II | | |
| ***ჯგუფი*** |  | | |
| ***გამოცდის ფორმა*** | ღია წიგნი | | |
| ***გამოცდის***  ***ხანგრძლივობა*** | 2 საათი | | |
| ***მაქსიმალური ქულა*** | 120 | | |
| ***სტუდენტის სახელი და გვარი:*** | | | |

სახელი: ქულა:

ფინალური 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 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. 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. **ვებ ბრაუზერში გახსენით მისამართი** [http://192.168.210.5](http://192.168.210.5/) **და ატვირთეთ არქივი. არქივი დატოვეთ დესკტოპზე**

**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.
5. თუ გაურკვეველი შეცდომა გაქვთ, ცადეთ მენიუდან Build > Clean Solution და Build > Clean Blank Project

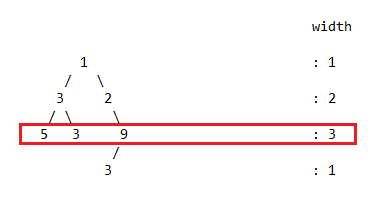
**ამოცანა 1. (60 ქულა) ხის მაქსიმალური სიგანე**

ხის სიგანე თითოეული სიღმისთვის(დონისთის) არის ამ სიღრმეზე არსებული ელემეტების რაოდენობა. ხის მაქსიმალური სიგანე არის, დონეების სიგანეებს შორის მაქსუმალური.

თქვენი ამოცანაა იპოვოთ ორობითი ხის მაქსიმალური სიგანე და დააბრუნოთ ხის შესაბამის დონეზე არსებული ელემენტების ჯამი. თუ მაქსიმალური სიგანე რამდენიმე დონეზე ფიქსირდება, დააბრუნეთ იმ დონის ელემენტების ჯამი, რომელიც უფრო ახლოსაა root ელემენტთან.

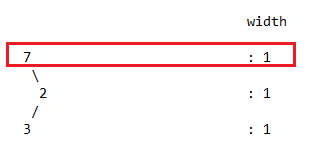
ქვემოთ მოცემულია რამდენიმე მაგალითი:

**მაგალითი 1**



**პასუხი: 17** (5 + 3 + 9)

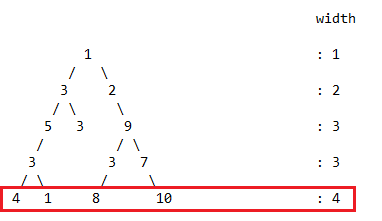
**მაგალითი 2**



**პასუხი: 7**

ამ შემთხვევაში ყველა დონეზე ერთი გვაქვს სიგანე, ამიტომ პასუხი არის 7, root თან ყველაზე ახლო დონეზე მყოფი ელემენტების ჯამი.

**მაგალითი 3**



**პასუხი: 23** (4 + 1 + 8 + 10)

დაწერეთ treeMaxWidth ფუნქციის რეალიზაცია. ფუნქციას გადაეცემა არანულოვანი ხე. ხეში შეიძლება იყოს 10^5 წვერო და თქვენი ამოხსნა უნდა მუშაობდეს O(n\*log(n)) დროში. ტესტების მაგალითები შეგიძლიათ იხილოთ **project\Blank Project\treewidth** ფოლდერში.

**ამოცანა 2. (60 ქულა) მასივში გადახტომა**

მოცემული გაქვთ მასივი არაუარყოფითი რიცხვებით. თამაშის წესი შემდეგია: თუ თქვენ იმყოფებით მასივში i პოზიციაზე, მაშინ შეგიძლიათ გადახვიდეთ i + arr[i] ან i - arr[i] პოზიციაზე, თუ რათქმაუნდა არ ცდება მასივის დიაპაზონს. საწყის ეტაპზე თქვენ იმყოფებით start პოზიციაზე, დაადგინეთ შესაძლებელია თუ არა მასივის ისეთ პოზიციაზე გადასვლა რომლის მნიშვნელობა არის 0.

დაწერეთ minJumps ფუნქციის რეალიზაცია:

bool minJumps (int[] arr, int n, int start);

ფუნქციას გადაეცემა მასივი, მასივში ელემენტების რაოდენობა და საწყისი პოზიცია. ფუნქცია აბრუნებს true ს თუ შესაძლებელია 0 მნიშვნელობის მქონე ელემენტზე გადასვლა და false წინააღმდეგ შემთვევაში.

**მაგალითი 1:**

Input: arr = [4,2,3,0,3,1,2], start = 5

Answer: true

Description: 5 -> 4(5 - arr[5]) -> 1(4-arr[4]) -> 3(1+arr[1])

**მაგალითი 2:**

Input: arr = [1,2,3,3,0,3,2,2], start = 0

Answer: true

Description: 0 -> 1(0 + arr[0]) -> 3(1+arr[1]) -> 6(3-arr[3]) -> 4(6-arr[6])

**მაგალითი 3:**

Input: arr = [3,0,2,1,2], start = 2

Answer: false

Description: ვერანაირად ვერ მოვახერხებთ arr[1] ზე მოხვედრას

მასივში ელემენტების რაოდენობა არ აღემატება 1000-ს. თქვენი ამოხსნა არ უნდა მუშაობდეს O(n\*n) ზე უფრო ცუდ დროში. ტესტების მაგალითები შეგიძლიათ იხილოთ **project\Blank Project\jump** ფოლდერში.

ერთ ფაილში რამდენიმე ტესტია გაერთიანებული. ფაილის პირველ ხაზში მოცემულია ტესტების რაოდენობა, შემდეგ მოდის შესაბამისი რაოდენობის ტესტი. თითოეული ტესტისთის პირველ ხაზში მოცემულია რიცხვების რაოდენობას, საწყის პოზიციას და პასუხი(1 ნიშნავს true ს ხოო 0 ნიშნავს false-ს). მეორე ხაზში მოცემულია რიცხვების სია.