**Praktiskais darbs**

**Programmatūras testēšana**

**„Naturālu skaitļu trijnieki”**

**Programmas testēšana**

Izstrādāja:

GRUPA grupas students

Vārds Uzvārds

Rīga

**Saturs**

[1. Funkcionālās diagrammas izmantošanas metode 3](#_Toc532676646)

[1.1. Funkcionālā diagramma 4](#_Toc532676647)

[1.2. Lēmumu tabula 5](#_Toc532676648)

[1.3. Testu kopums programmatūras testēšanai 6](#_Toc532676649)

[2. Baltās kastes kombinatoriskā nosacījumu pārklājuma metode 7](#_Toc532676650)

[3. Testēšanas rezultāti 10](#_Toc532676651)

[4. Bibliogrāfija 11](#_Toc532676652)

# Funkcionālās diagrammas izmantošanas metode

Programmatūras testēšanai izvēlējos funkcionālās diagrammas izmantošanas metodi, jo tā ļauj izpētīt arī ieejas nosacījumu kombinācijas. Lai izveidotu testu kopumu vispirms, ir jānosaka cēloņi un sekas. Izvēlētā programma atradīs tādus naturālu skaitļu trijniekus a, b, c, lai izpildītos nosacījums: a <= b <= c <= n, kur a2+b2=c2.

**Cēloņi:**

1. Ir ievadīta vērtība

2. Ievadītā vērtība ir skaitlis

3. Ievadītā vērtība ir naturāls skaitlis

4. Ievadītā vērtība ir skaitlis, kas ir lielāks par 4

**Sekas:**

6. Kļūdas paziņojums par nederīgu simbolu

7. Kļūdas paziņojums par to, ka skaitlis nav naturāls

8. Kļūdas paziņojums par to, ka nav tādu vērtību, lai izpildītos programmas nosacījums

9. Programma veiksmīgi atgriež rezultātu

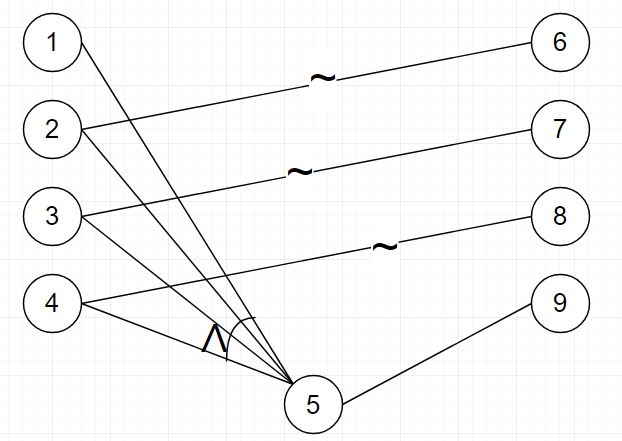
Lai atvieglotu diagrammas izveidi, izdalu palīgsekas.

**Palīgsekas:**

5. Ievadītā vērtība ir korekta

## Funkcionālā diagramma

No sekām un cēloņiem tālāk izveidoju funkcionālo diagrammu, kas ir redzama 1.1. attēlā.



1.1. att. Funkcionālā diagramma

## Lēmumu tabula

No funkcionālās diagrammas izveidoju lēmumu tabulu (), kuru tālāk varēs izmantot testu kopuma izveidei.

1.1. tabula

Lēmumu tabula

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | 1 | 2 | 3 | 4 |
| C  ē  l  o  ņ  i | 1 | 1 | 1 | 1 | 1 |
| 2 | 0 | 1 | 1 | 1 |
| 3 |  | 0 | 1 | 1 |
| 4 |  |  | 0 | 1 |
| S  e  k  a  s | 6 | 1 |  |  |  |
| 7 |  | 1 |  |  |
| 8 |  |  | 1 |  |
| 9 |  |  |  | 1 |

## Testu kopums programmatūras testēšanai

Izmantojot lēmumu tabulu konstruēju testu kopumu programmatūras testēšanai

1.2. tabula

Testu kopums

|  |  |  |
| --- | --- | --- |
| Testa nr. | Ieeja (cēloņi) | Izeja (sekas) |
| 1 | a | Tas nav skaitlis, ievadiet naturālu skaitli! |
| 2 | -123 | Skaitlim ir jābūt naturālam skaitlim, ievadiet naturālu skaitli! |
| 3 | 3 | Nav tādu vērtību, ievadiet citu skaitli! |
| 4 | 10 | 3 4 un 5; 6 8 un 10 |

Papildināsim tabulu ar 5. un 6. testiem lai pārbaudīt robežvērtības 4. cēlonim

1.3. tabula

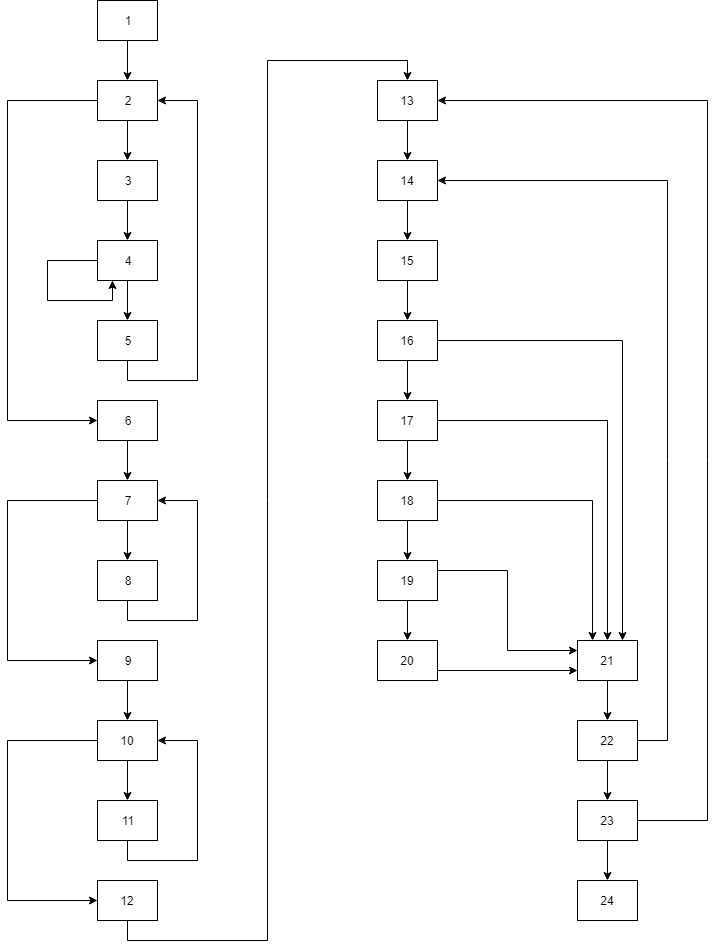
|  |  |  |
| --- | --- | --- |
| Testa nr. | Ieeja (cēloņi) | Izeja (sekas) |
| 1 | a | Tas nav skaitlis, ievadiet naturālu skaitli! |
| 2 | -123 | Skaitlim ir jābūt naturālam skaitlim, ievadiet naturālu skaitli! |
| 3 | 3 | Nav tādu vērtību, ievadiet citu skaitli! |
| 4 | 10 | 3 4 un 5; 6 8 un 10 |
| 5 | 4 | Nav tādu vērtību, ievadiet citu skaitli! |
| 6 | 5 | 3 4 un 5 |

# Baltās kastes kombinatoriskā nosacījumu pārklājuma metode

Programmas pirmteksts:

|  |  |
| --- | --- |
| 1 | int main() |
|  | { |
|  | float n; |
|  | float c; |
|  | cout << " " << endl; |
|  | cout << "Ievadiet naturalu skaitli" << endl; |
|  | cout << "Programma atradis tadus naturalu skaitlu trijniekus a, b, c," << endl; |
|  | cout << "lai izpilditos nosacijums: a <= b <= c <= n, kur a^2+b^2=c^2" << endl; |
|  | cin >> n; |
| 2 | while(!cin) |
|  | { |
| 3 | cout << "Tas nav skaitlis, ievadiet naturalu skaitli!" << endl; |
|  | cin.clear(); |
| 4 | while (cin.get() != '\n'); |
| 5 | cin >> n; |
| 6 | } |
| 7 | while(n <= 0) |
|  | { |
| 8 | cout << "Skatilim ir jabut naturalam skaitlim, ievadiet naturalu skaitli!" << endl; |
|  | cin >> n; |
| 9 | } |
| 10 | while(int(n) <= 4) |
|  | { // Programma izvadis rezultatus tikai tad, ja n ir vismaz 5, jo tad izvaditie skaitli atbildis prasibam |
| 11 | cout << "Nav tadu vertibu, ievadiet citu skaitli!" << endl; |
|  | cin >> n; |
| 12 | } |
| 13 | for (int a = 1; a <= n; a++) |
|  | { |
| 14 | for (int b = 1; b <= n; b++) |
|  | { |
| 15 | c = sqrt(a\*a + b\*b); |
| 16,17,18,19 | if (int(c) == c && a <= b && b <= c && c <= n) |
|  | { |
| 20 | cout << a << " " << b << " un " << c << endl; |
| 21 | } |
| 22 | } |
| 23 | } |
| 24 | system("PAUSE"); |
|  | return 0; |
|  | } |

Balstoties uz pirmtekstu izveidoju grafu, kas uzskatāmi attēlo funkcijas struktūru:



2.1. att. Funkcijas struktūras grafs

Grafa neatkarīgo ceļu skaitu nosaka ciklomatiskā sarežģītība, kuru aprēķina pēc formulas C=L -V + 2\*P (izeja nav saistīta ar ieeju). Izveidotajam grafam C = 33-24+2\*1 = 11

Maršruti:

1. 1,2,6,7,9,10,12-24
2. 1,2,3,4,4,5, 2,6,7,9,10,12-24
3. 1-5,2,6,7,9,10,12-24
4. 1,2,6,7,8,7,9,10,12-24
5. 1,2,6,7,9,10,11,10,12-24
6. 1,2,6,7,9,10,12,13,14,15,16,21-24
7. 1,2,6,7,9,10,12,13,14,15,16,17,21-24
8. 1,2,6,7,9,10,12,13,14,15,16,17,18,21-24
9. 1,2,6,7,9,10,12,13,14,15,16,17,18,19,21-24
10. 1,2,6,7,8,7,9,10,12-22,14-24
11. 1,2,6,7,8,7,9,10,12-23,13-24
12. Maršrutu pārklāj 4. un 6. testi
13. Maršrutu pārklāj 1. tests
14. Maršrutu pārklāj 1. tests
15. Maršrutam izveidojam 7. un 8. testus
16. Maršrutam izveidojam 9. un 10. testus
17. Maršrutu pārklāj 4. un 6. testi
18. Maršrutu pārklāj 4. un 6. testi
19. Maršrutu pārklāj 4. un 6. testi
20. Maršrutu pārklāj 4. un 6. testi
21. Maršrutu pārklāj 4. un 6. testi
22. Maršrutu pārklāj 4. un 6. testi

2.1. tabula

|  |  |  |
| --- | --- | --- |
| Testa nr. | Ieeja (cēloņi) | Izeja (sekas) |
| 1 | a | Tas nav skaitlis, ievadiet naturālu skaitli! |
| 2 | -123 | Skaitlim ir jābūt naturālam skaitlim, ievadiet naturālu skaitli! |
| 3 | 3 | Nav tādu vērtību, ievadiet citu skaitli! |
| 4 | 10 | 3 4 un 5; 6 8 un 10 |
| 5 | 4 | Nav tādu vērtību, ievadiet citu skaitli! |
| 6 | 5 | 3 4 un 5 |
| 7 | -123 un 8 | 3 4 un 5 |
| 8 | -123 un 'b' | Programmas kļūda |
| 9 | 3 un 10 | 3 4 un 5; 6 8 un 10 |
| 10 | 3 un 'b' | Programmas kļūda |

Tagad izstrādātais testu kopums ir pietiekams programmas testēšanai.

# Testēšanas rezultāti

Izmantojot izveidoto testu kopumu veicu testēšanu, kā rezultāti ir redzami 3.1 tabulā.

3.1. tabula

Testēšanas rezultāti

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Testa nr. | Ieeja (cēloņi) | Paredzētais rezultāts | Iegūtais rezultāts | Piezīmes |
| 1 | a | Kļūda par to, ka nav ievadīts skaitlis. | atbilst | Programma turpina darbu, līdz nav ievadīta vērtība, kas sniedz pozitīvu rezultātu. |
| 2 | -123 | Kļūda par to, ka nav ievadīts naturāls skaitlis. | atbilst | Programma turpina darbu, līdz nav ievadīta vērtība, kas sniedz pozitīvu rezultātu. |
| 3 | 3 | Paziņojums par to, ka nav tādu vērtību. | atbilst | Programma turpina darbu, līdz nav ievadīta vērtība, kas sniedz pozitīvu rezultātu. |
| 4 | 10 | 3 4 un 5; 6 8 un 10 | atbilst | Programma izvada atbildi un beidz darbu |
| 5 | 4 | Paziņojums par to, ka nav tādu vērtību. | atbilst | Programma turpina darbu, līdz nav ievadīta vērtība, kas sniedz pozitīvu rezultātu. |
| 6 | 5 | 3 4 un 5 | atbilst | Programma izvada atbildi un beidz darbu |
| 7 | -123 un 8 | 3 4 un 5 | atbilst | Programma izvada atbildi un beidz darbu |
| 8 | -123 un 'b' | Programmas kļūda | neatbilst | Programma ieciklojas jo atkārtoti ievadīta vērtība nav skaitlis |
| 9 | 3 un 10 | 3 4 un 5; 6 8 un 10 | atbilst | Programma izvada atbildi un beidz darbu |
| 10 | 3 un 'b' | Programmas kļūda | neatbilst | Programma ieciklojas jo atkārtoti ievadīta vērtība nav skaitlis |

Kā redzams pēc testēšanas rezultātiem, programmā ir atrasti divi defekti kas saistīti ar ievades apstrādi.

# Bibliogrāfija