Картина, която съдържа текст, графична колекция

Описанието е генерирано автоматичноТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ВАРНА

Факултет по изчислителна техника и автоматизация

Катедра „Софтуерни и интернет технологии“

**СЕМЕСТРИАЛНА ДОМАШНА РАБОТА**

по дисциплината „Базово програмиране”

на тема: „Конкурс за красота”

Вариант 26

|  |  |
| --- | --- |
| Изготвил: Ферай Фахриев Саидов | Проверил: |
| Специалност: СИТ |  |
| Група: Iб |  |
| Факултетен номер: 22621706 |  |

2023

Съдържание

[I. Задание на проекта 3](#_Toc120821980)

[II. Анализ на решението 4](#_Toc120821981)

[1. Структура за данните в програмата 4](#_Toc120821982)

[2. Реализация на условие A 4](#_Toc120821983)

[3. Реализация на условие B 4](#_Toc120821984)

[4. Реализация на условие C 5](#_Toc120821985)

[5. Реализация на условие D 5](#_Toc120821986)

[6. Реализация на условие E 6](#_Toc120821987)

[7. Реализация на условие F 7](#_Toc120821988)

[8. Реализация на условие … - допълнение първо 7](#_Toc120821989)

[9. Реализация на условие … - допълнение второ 8](#_Toc120821990)

[10. Реализация на допълнение трето 8](#_Toc120821991)

[III. Упътване за употреба 9](#_Toc120821992)

[1. Впишете съответната част от проекта 9](#_Toc120821993)

[2. Впишете съответната част от проекта 9](#_Toc120821994)

[IV. Примерно действие на програмата 10](#_Toc120821995)

[1. Условие A 10](#_Toc120821996)

[2. Условие B 10](#_Toc120821997)

[3. Условие C 10](#_Toc120821998)

[4. Условие D 10](#_Toc120821999)

[5. Условие E 10](#_Toc120822000)

[6. Условие F 11](#_Toc120822001)

[7. Допълнение първо 11](#_Toc120822002)

[8. Допълнение второ 11](#_Toc120822003)

[9. Допълнение трето 11](#_Toc120822004)

Задание на проекта

Да се напише компютърна програма, реализираща информационна система, която поддържа конкурс за красота (номер на участник, име, възраст, пол, бедрена обиколка, раменна обиколка, обиколка на шията, обиколка на прасеца). Програмата съхранява и обработва данни за участниците в конкурса. Максималния брой участници е 30. Базова задача A. Меню за избор на функциите от програмата Функции от програмата са: B. Добавяне: a. Добавянето трябва да позволява да се въвеждат различен брой нови участници като се допълват в масива, като не трябва да превишават максималния брой (30). Пример: Добавяне на списък с участници. Въвежда се цяло число n, което позволява да се въведат n на брой участници. n не може да надвишава свободните елементи в масива. C. Извеждане на екрана a. Извеждане на всички участници в оформен вид D. Търсене и извеждане на екрана: a. Намиране и извеждане на всички участника с най-ниска възраст b. Търсене на участник по име E. Подреждане на основния масив, без да се извежда на екрана: a. Подреждане на участниците по нарастващ ред на възрастта им F. Управление на файл: a. Извеждане на масива във файл (двоичен) b. Въвеждане на масива от файл (двоичен) Допълнение първо (+ базова задача) G. Създайте подменю, в което се влиза от основното, с нови функции за: a. Извеждане на участниците в подреден ред по години и име, без да се променя основния масив. b. Търсене и извеждане на участниците по въведена възраст и пол Допълнение второ (+ базова задача) H. Разделяне на участниците в категории (10-11 седмица) a. Разпределят се участниците във възрастови категории I (14-16), II (17-19), III (20-22), IV (23-25), всяка категория в отделен нов масив. b. Извеждат се списъците на участниците от всяка категория, ако в дадена категория няма участници се извежда съобщение за това; I. Определяне на победителите от всяка категория: (9-10 седмица) c. Победител в конкурса се определя по формулата за пропорционални телесни мерки (П) – П = Б/(Ш+Н+Р) (бедрена обиколка (Б), раменна обиколка (Р), обиколка на шията (Ш), обиколка на прасеца (Н)); d. Участниците с П между 0,54 и 0,62 се подреждат по намаляващ ред, за да се определи победителя; e. Участниците с П повече от 0,63 се нареждат в нарастващ ред, преди участниците с П по-малко от 0,53 който се подреждат в нарастващ ред; Допълнение трето (+ базова задача) J. Данните в програмата да се попълват автоматично от файл при стартиране и да се записват автоматично във файл при затваряне на програмата.

Анализ на решението

Структура за данните в програмата

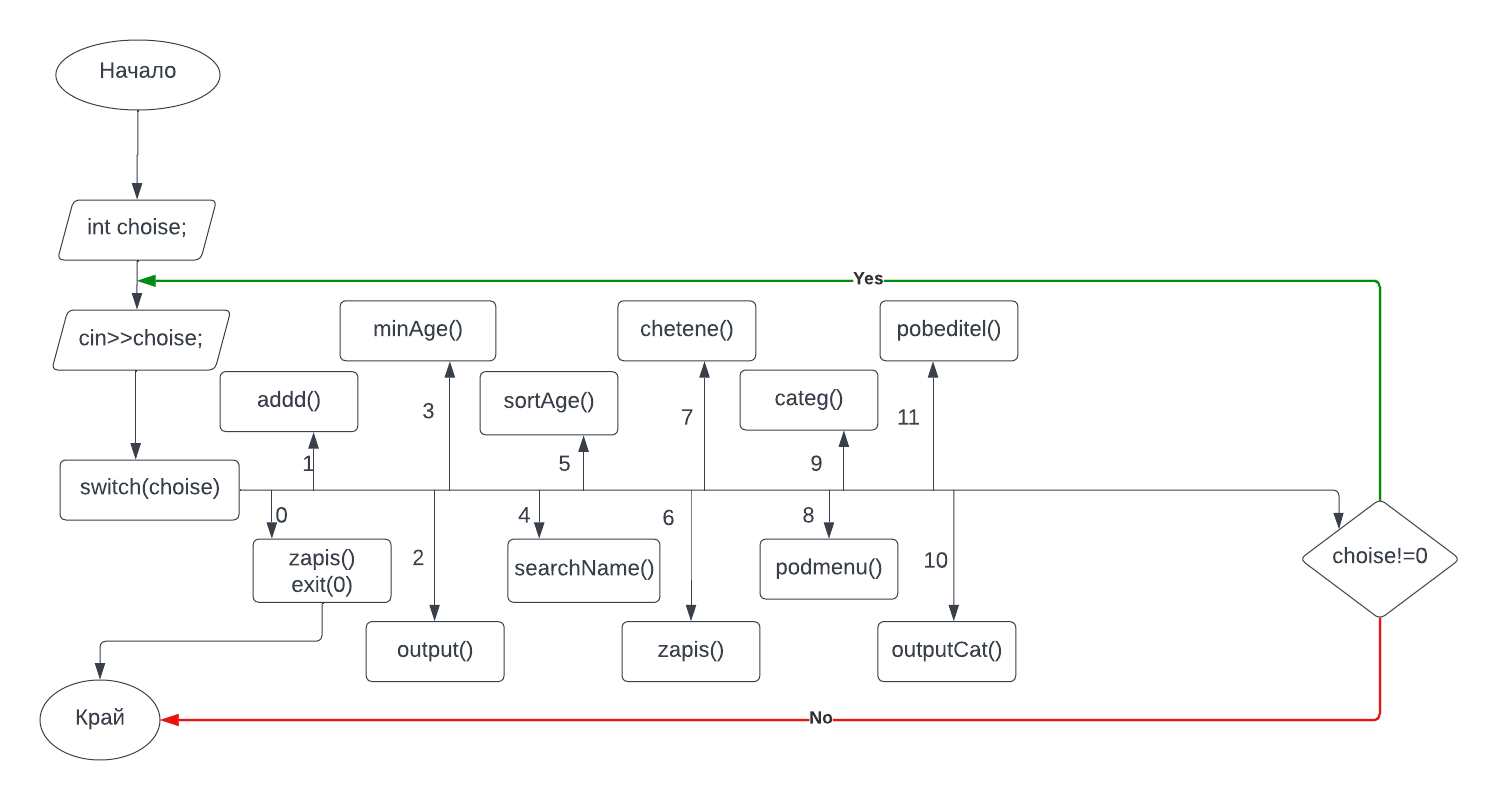
|  |  |  |
| --- | --- | --- |
| Структура | Обяснение | Примерени стойности |
| struct member {  int number;  char name[50];  int age;  char sex;  double bedr\_o;  double ram\_o;  double shiq\_o;  double pras\_o;    }; | * Number – съхранява номер на участник * Name[50] – съхранява името на участника, което не трябва да надвишава 50 символа. * Age- съхранява стойност за възрастта на участника * sex- съхранява символ, който отговаря на пола на участника * bedr\_o – съхранява стойност за бедрената обиколка на участника * ram\_o – съхранява стойност за раменната обиколка на участника * shiq\_o - съхранява стойност за обиколката на шията на участника * pras\_o \_ съхранява стойност за обиколката на прасеца на участника | 1  Радослав Петров  19  M  49  48  12  15 |

Реализация на условие A

Анализ на алгоритъма, който трябва да се реализира

1. Извеждат се на екрана възможните опции от менюто.
2. Програмата чака въвеждане на цяло число, което отговаря на една от изведените опции
3. Изпълнява се избраната от потребителя опция, след което отново се извеждат на екрана възможните опции, освен ако не се въведе 0, при което менюто се затваря и програмата се затваря.

Блок схема на алгоритъма



Функция, с която е реализиран алгоритъма

void menu(member arr[], int& len, member gr1[], member gr2[], member gr3[], member gr4[], int& len1, int& len2, int& len3, int& len4);

След извикване, на екрана се извеждат възможните опции за избор на клиента. След извеждане на екрана функцията чака въвеждане от клавиатурата на цяло число, отговарящо на номер от изведените на екрана опции, като след въвеждане, се изпълнява избора на клиента. След изпълняване на избора, на екрана отново се извеждат опциите на клиента до въвеждане на номера за изход (0), като при въвеждане на 0, цикълът се прекратява и програмата се затваря.

#### Входни данни на функцията

Функцията menu() приема като параметри масива, в който искаме да въведем участниците от групата (member arr[]), променлива, която отговаря на броя на елементите в него (int& len), който се променя, при въвеждане на нови стойности в масива, 4 отделни масива за четирите групи (member gr1[], member gr2[], member gr3[], member gr4[]), на които ще се раздели основния масив и 4 променливи, отговарящи на броя на елементите във всяка от групите, които са зададени като параметър, понеже не знаем колко елемента ще съдържа всяка група (int& len1, int& len2, int& len3, int& len4).

#### Изходни данни на функцията или данни, които се извеждат

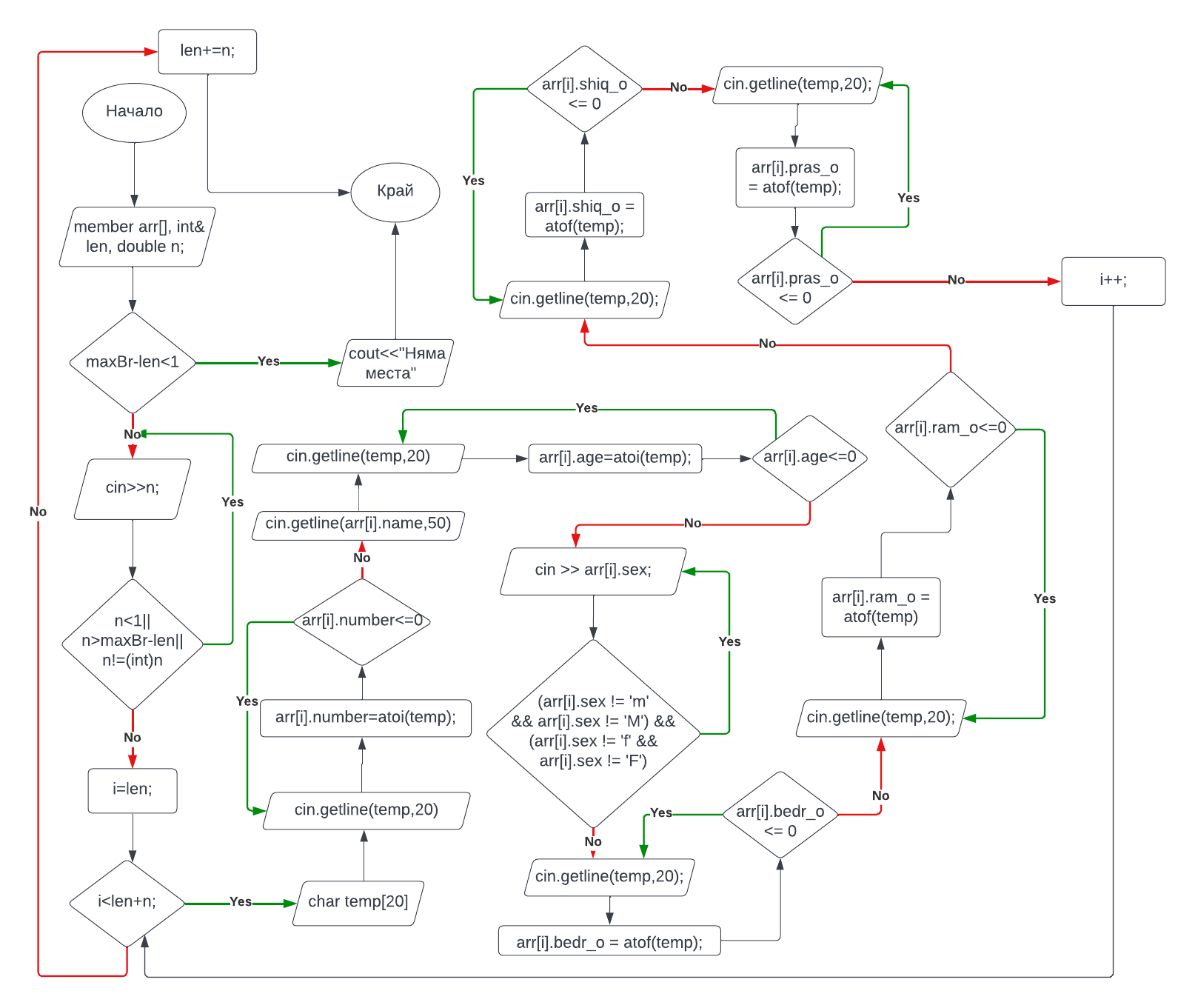
Функцията не връща резултат, а представлява препратка към други функции. Тя извежда на екрана възможни избори на клиента и чака да се въведе число, отговарящо на избора. При въвеждане на числото, функцията ще изведе на екрана резултата от избраната функция. Функцията е от тип void, понеже не връща резултат, а само извежда менюто на екрана и чака въвеждане на стойност.

Реализация на условие B

Анализ на алгоритъма, който трябва да се реализира

1. Въвежда се променлива, цяло число, отговаряща на броя участници, които ще се въведат в масива, след което се добавят от клавиатурата данните за всеки един участник. След това към броя участници, до момента въведени в масива се добавя броя на новодобавените участници.

Блок схема на алгоритъма



Функция, с която е реализиран алгоритъма

void addd(member arr[], int& len);

Функцията въвежда данни в масив от участници (member) и определя размера му, който не трябва да надвишава максималния (maxBr = 30). При изпълнение на функцията се въвежда число, което отговаря на броя на елементите, които ще се въведат в масива. Този брой, добавен към броя на елементите, вече в масива не трябва да надвишава максималния (30). След въвеждане на броя на елементите, които ще се въведат, започва самото въвеждане, което представлява цикъл, като при всяка една итерация се въвеждат стойности за всеки един участник – номер, име, възраст и т.н. Въведеното число като брой на елементите, които ще се въвеждат се добавя към броя на елементите вече в масива и именно затова параметъра за брой на елементите се предава по адрес, за да може и фактическия да се промени, с която промяна можем да следим при следващо въвеждане дали броя на елементите няма да надвиши максималния.

#### Входни данни на функцията

Входните данни са масив от участници (member arr[]), в който ще въвеждаме участници и брой на елементите в дадения масив (int& len). Броя на елементите се предава по адрес, понеже при всяко едно въвеждане той се променя и се проверява дали могат да се въведат допълнително участници.

#### Изходни данни на функцията или данни, които се извеждат

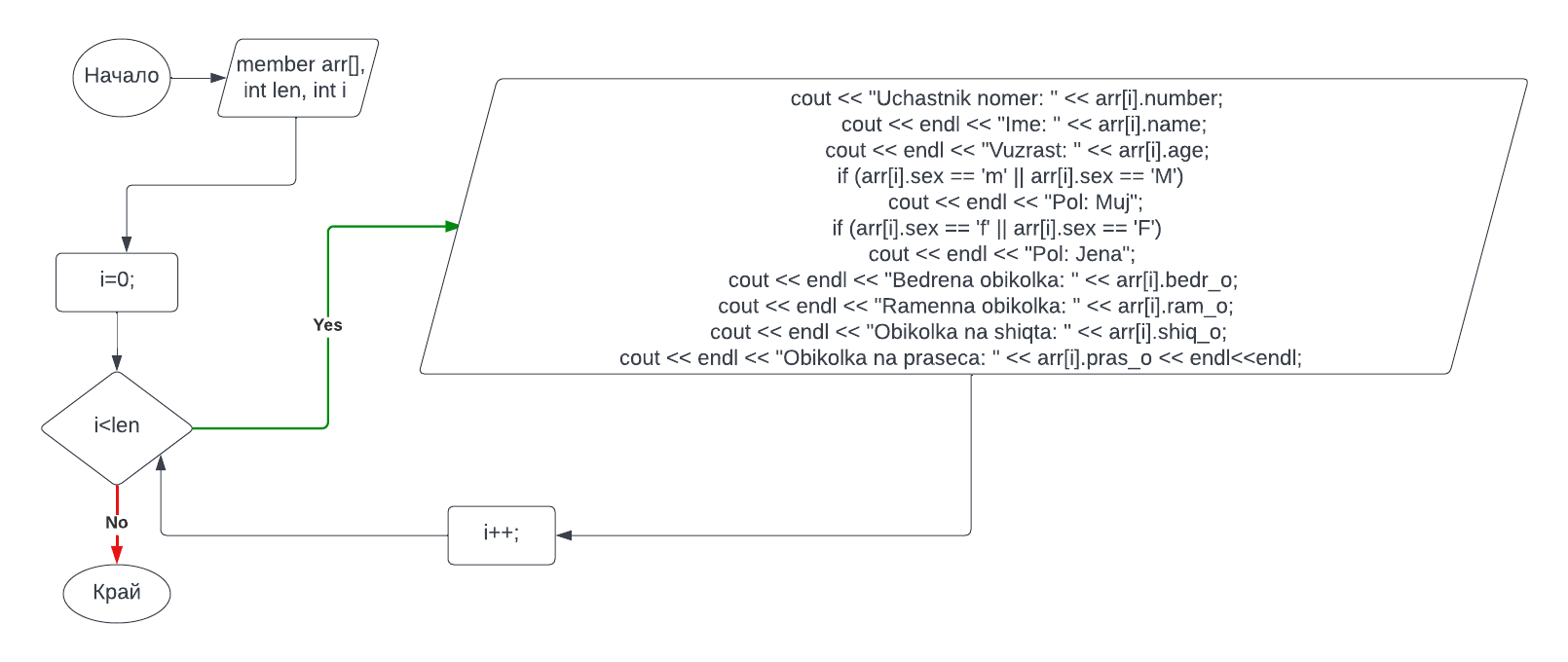
Функцията няма да изведе нищо на екрана и именно затова е тип void. Тя въвежда елементи в масива, зададен като параметър и променя стойността на броя на елементите в него, спрямо избора на клиента.

Реализация на условие C

Анализ на алгоритъма, който трябва да се реализира

Върти се цикъл, докато брояча не стане равен на броя на участниците в масива и при всяка итерация, се извеждат всички данни на участника с индекс, равен на стойността на брояча до момента.

Блок схема на алгоритъма



Функция, с която е реализиран алгоритъма

void output(member arr[], int len);

Функцията извежда на екрана елементи, от масива от участници, въведен като параметър, равен на броя на елементите (len), отново предаден като параметър. Тя представлява цикъл, като при всяка итерация се извеждат всички данни на участника, отговарящ на индекса, на който се намираме.

#### Входни данни на функцията

Входните данни на функцията са масив от участници (member arr[]) и брой на елементите в него (int len). От масива ще се извеждат всички данни, а броя на елементите се използва за край на цикъла, който ще се върти за да се изведе всеки един участник, тоест цикъла ще продължава докато брояча не стане равен на броя на елементите (len).

#### Изходни данни на функцията или данни, които се извеждат

При изпълнение на функцията ще се изведат като списък всички елементи от масива с участници, като при въведен пол ‘M‘ или ‘m‘ , на екрана се извежда “Пол: Мъж“, а при въведен пол ‘F’ или ‘f’ , се извежда „Пол: Жена“.

Реализация на условие D

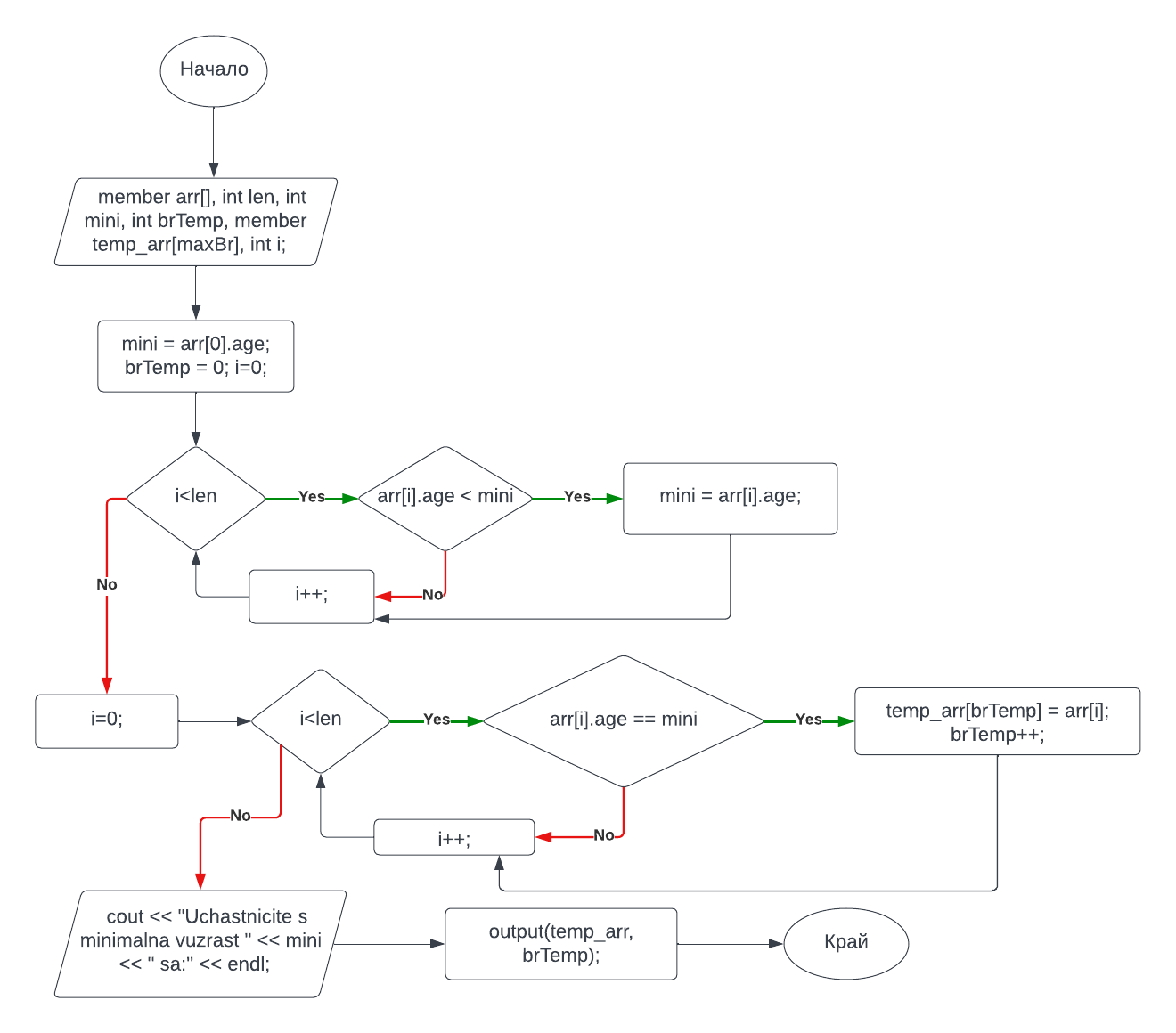
Анализ на алгоритъма, който трябва да се реализира

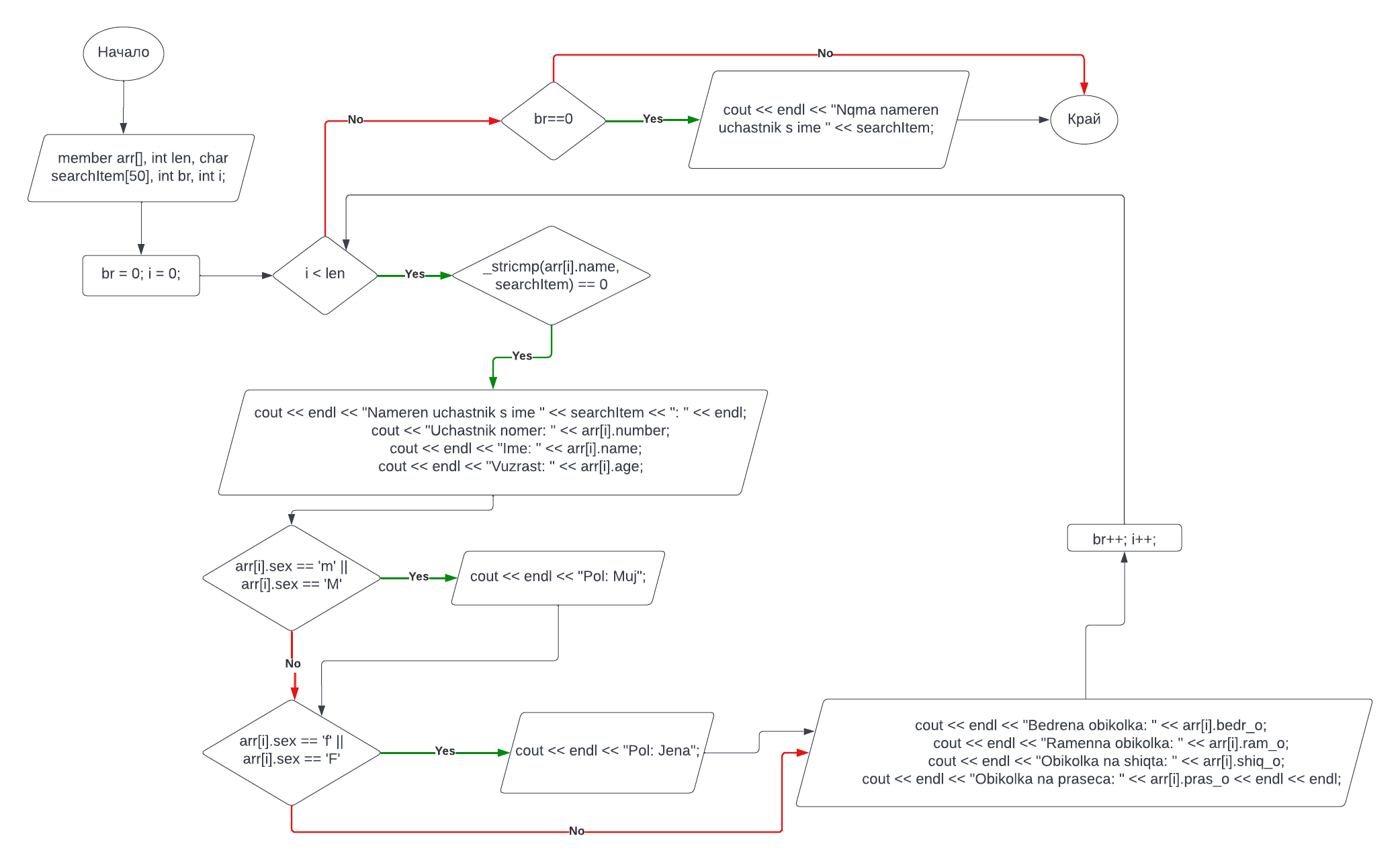
а. С помощта на цикъл се намира минималната възраст в масива от участници, след което се извеждат всички участници с такава възраст.

b. След въвеждане на търсеното име, масивът се обхожда и се въвеждат в отделен масив участниците с име, съвпадащо с името, което е въведено от клавиатурата, след което се извежда този допълнителен масив.

Блок схема на алгоритъма

Тук се поставя блок схема на описания алгоритъм.

minAge()

searchName()

Функция с която е реализиран алгоритъма

void minAge(member arr[], int len);

Функцията minAge, намира минималната възраст в масива от участници, след което въвежда елементите с тази възраст във втори масив, който не се подава като параметър и извежда всички данни от този нов масив на екрана, с помощта на функцията output.

void searchName(member arr[], int len);

Функцията searchName, търси и извежда участниците с име, което се въвежда допълнително, по време на изпълнение на функцията.

#### Входни данни на функцията

void minAge(member arr[], int len);

Входните данни във функцията minAge са масив от участници и броя на елементите в него, като от масива ще се търси минималната възраст, а броя на участниците се използва за условие на цикъла, с който се търси тази възраст.

void searchName(member arr[], int len);

Входните данни във функцията searchName са масив от участници и брой на елементите в него, като от масива ще се търси елемент с име, което се въвежда по време на изпълнение на функцията, а броя на елементите се използва за условие в цикъла, с който се обхожда масива.

#### Изходни данни на функцията или данни, които се извеждат

И двете функции са от тип void, понеже не връщат резултат, а само извеждат данни на екрана.

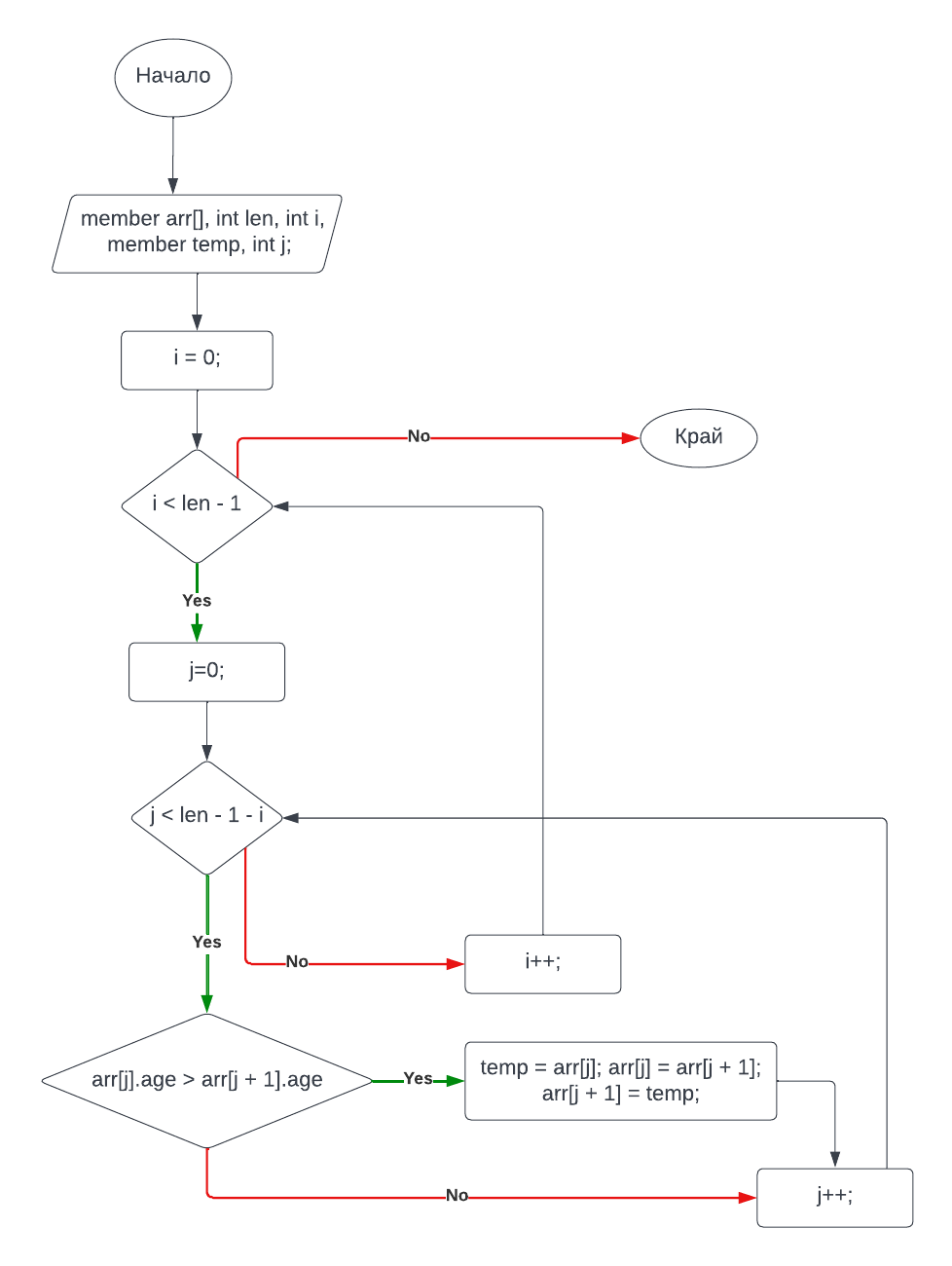
Функцията minAge ще изведе на екрана данните на участниците с най-малка възраст от масива, а searchName ще изведе участника с име, отговарящо на въведеното.

Реализация на условие E

Анализ на алгоритъма, който трябва да се реализира

1. Обхожда се масива с 2 цикъла, като първия отговаря на вече подредените елементи в него, а втория отговаря на елемента, който „слагаме на мястото му“, като при всяко обхождане на втория цикъл, сравняваме елемента с този, който се намира след него и ако е по-голям, им разменяме местата, след което първия цикъл се актуализира и повтаряме размяната, докато всички елементи не се подредят на местата си по нарастващ(възходящ) ред.

Блок схема на алгоритъма

sortAge()

Функция с която е реализиран алгоритъма

void sortAge(member arr[], int len);

Функцията подрежда участниците в масива по нарастващ ред на възрастта им. След употребата и, масивът въведен като параметър се подрежда, без да се извежда на екрана. Нищо не се извежда на екрана, а в паметта, данните в масива разменят местата си, ако е нужно.

#### Входни данни на функцията

Функцията приема като параметри масив от участници и брой на участниците в него, като в масива данните ще се сортират, а броя го ползваме за край на цикъла, който се използва за сортиране.

#### Изходни данни на функцията или данни, които се извеждат

Функцията не връща резултат и не извежда нищо на екрана, а само сортира въведения като параметър масив.

Реализация на условие F

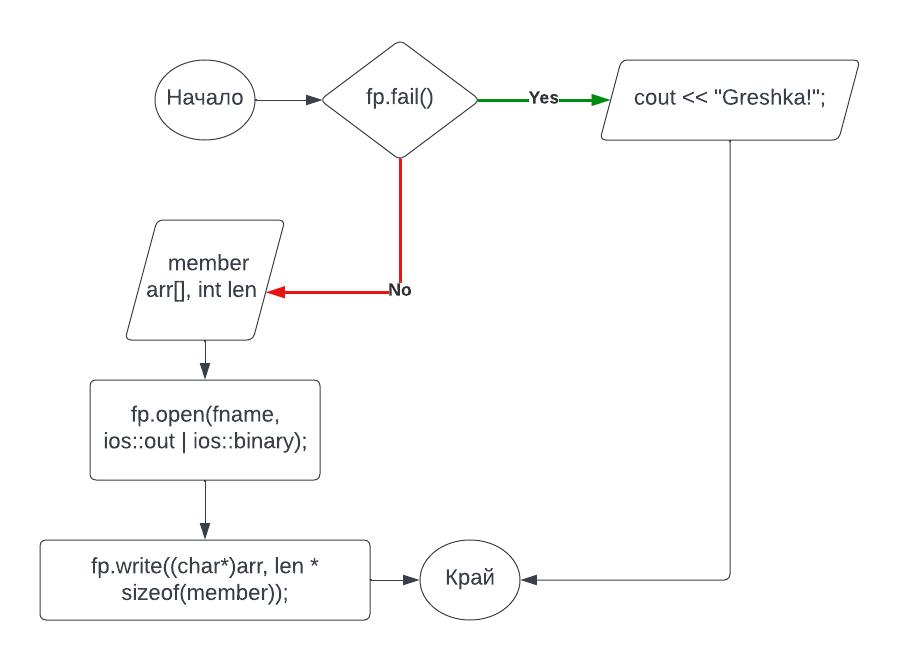
Анализ на алгоритъма, който трябва да се реализира

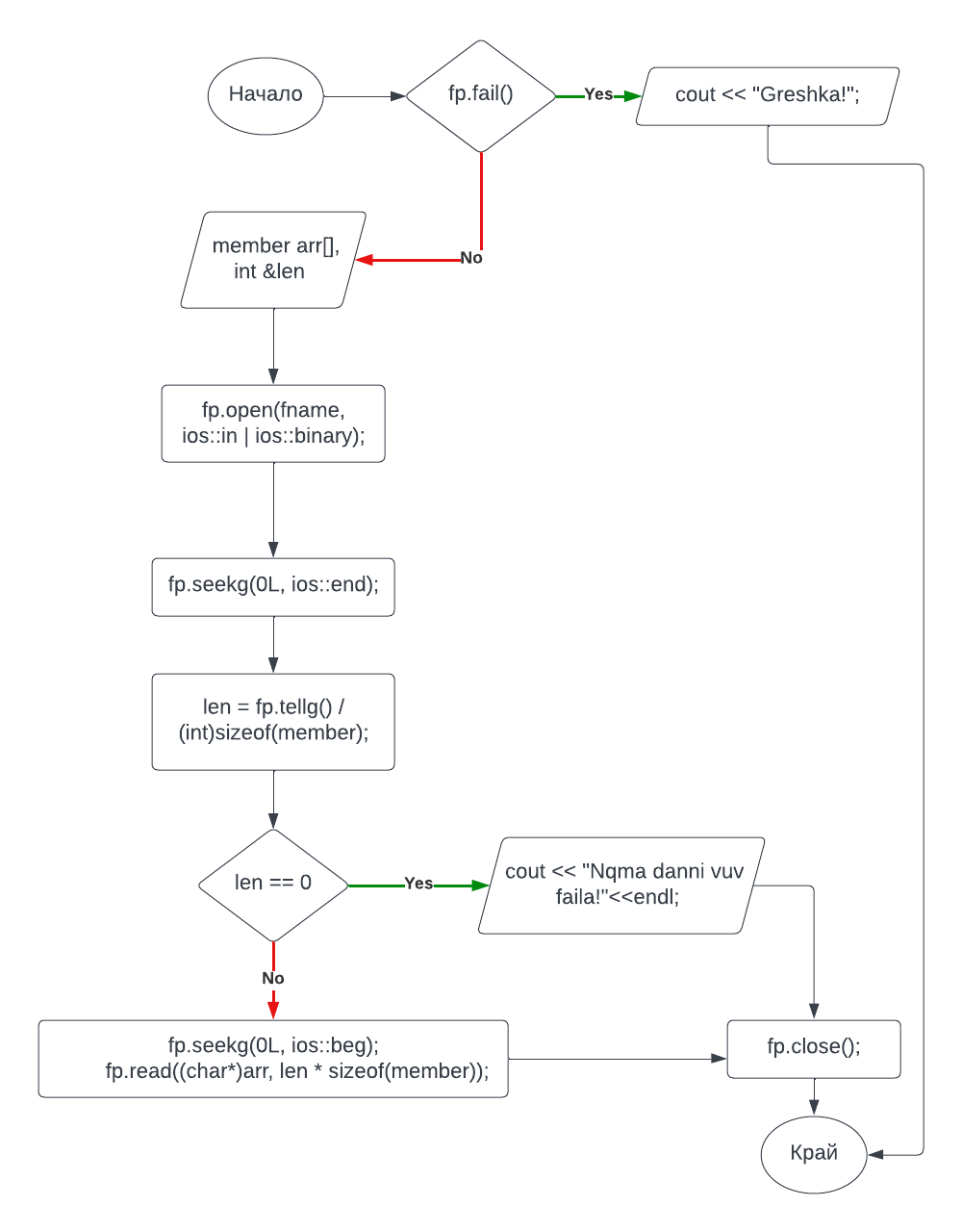
а. След отварянето на двоичен файл, се записва целия масив в него, като се запишат байтове, равни на броя на елементите в масива от участници, умножен по размера на масива в байтове.

b. Отваря се файл за четене и указателят застава в края му, след което се определят общия брой байтове от началото на файла до мястото на указателя (края на файла), след което този общ брой се разделя на размера на масива в байтове, което определя броя на елементите в масива, който се записва в променливата len, която се предава по адрес. След това указателят застава в началото на файла и се записват всички данни от него в масива(всички данни във файла са равни на броя на участниците, умножен по размера на структурата от участници).

Блок схема на алгоритъма

zapis()



chetene()

Функция с която е реализиран алгоритъма

void zapis(member arr[], int len);

Функцията zapis() записва данните от масива, въведен като параметър в двоичен файл. След извикване тя отваря двоичен файл за запис и записва всички данни от масива в него. Понеже при всяко затваряне на програмата, данните автоматично се записват в двоичния файл, то при отваряне, всички данни се изтриват във файла и се записват всички данни от масива в него, така че няма как да се изгубят данни.

void chetene(member arr[], int& len);

Функцията chetene() прочита данни от двоичен файл и ги записва в масива предаден като параметър, също намира броя на данните и ги записва в променливата (len).

#### Входни данни на функцията

Функциите приемат като параметри масив от участници, от който ще записват данни във файл, или в който ще записват данни от файл. Също приемат като параметър брой на участниците в масива, като по време на четене, този брой се променя, спрямо броя на прочетените елементи от файла, а по време на запис, тази променлива се използва за функцията write, като и казва колко пъти да вземе размера на структурата от паметта.

#### Изходни данни на функцията или данни, които се извеждат

Функциите не извеждат нищо на екрана и не връщат резултат и затова са от тип void.

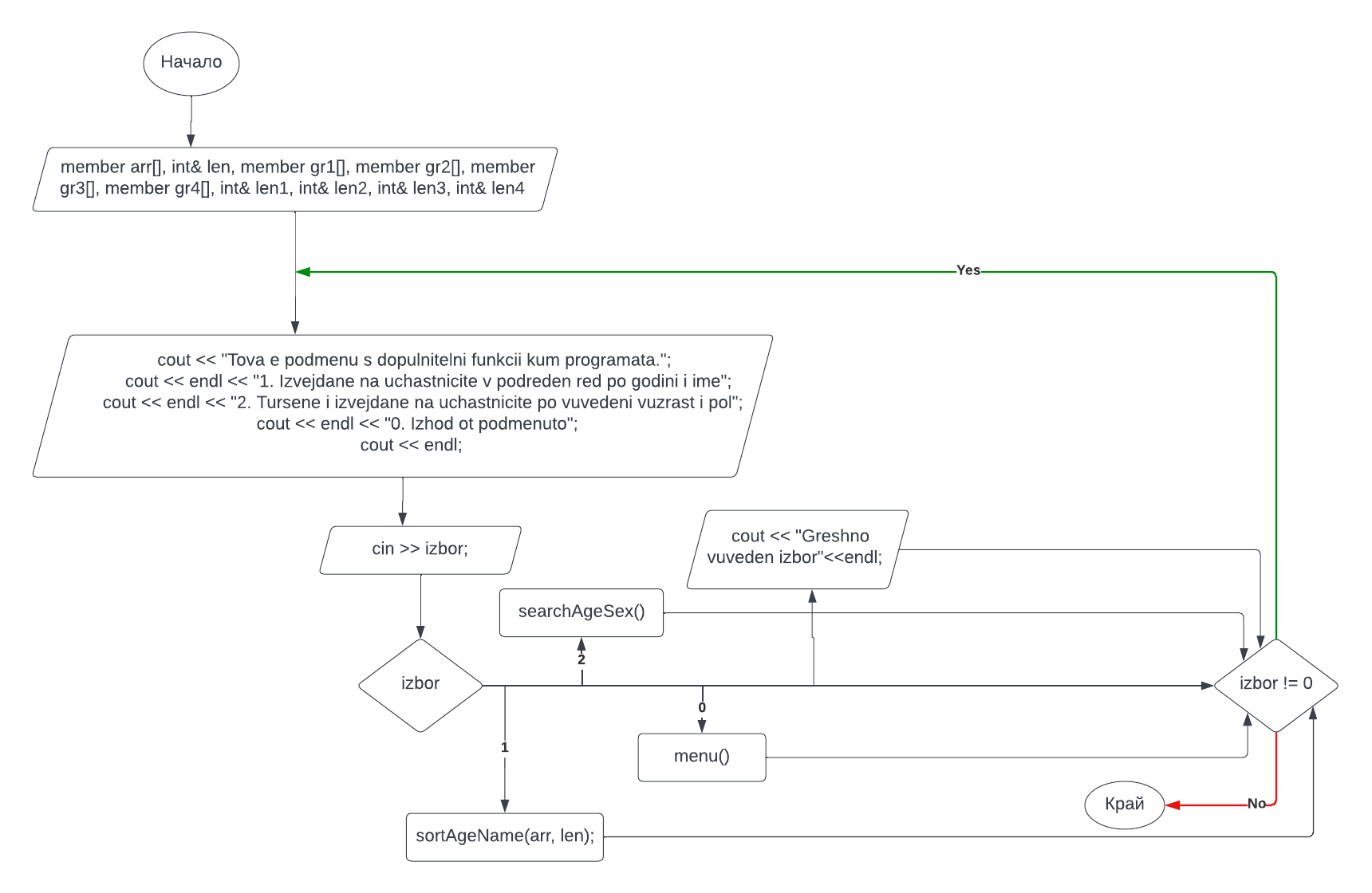
Реализация на условие … - допълнение първо

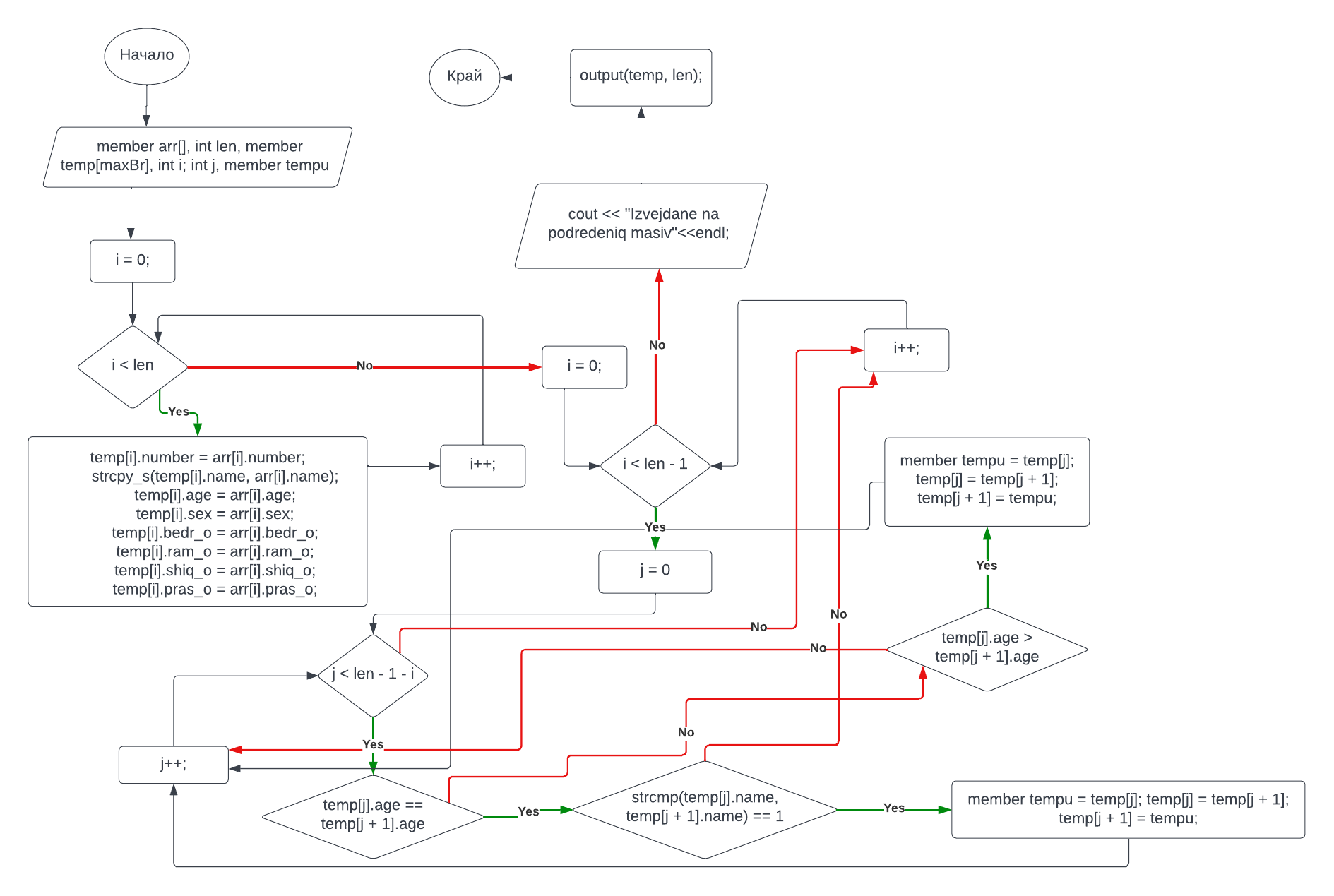
Анализ на алгоритъма, който трябва да се реализира

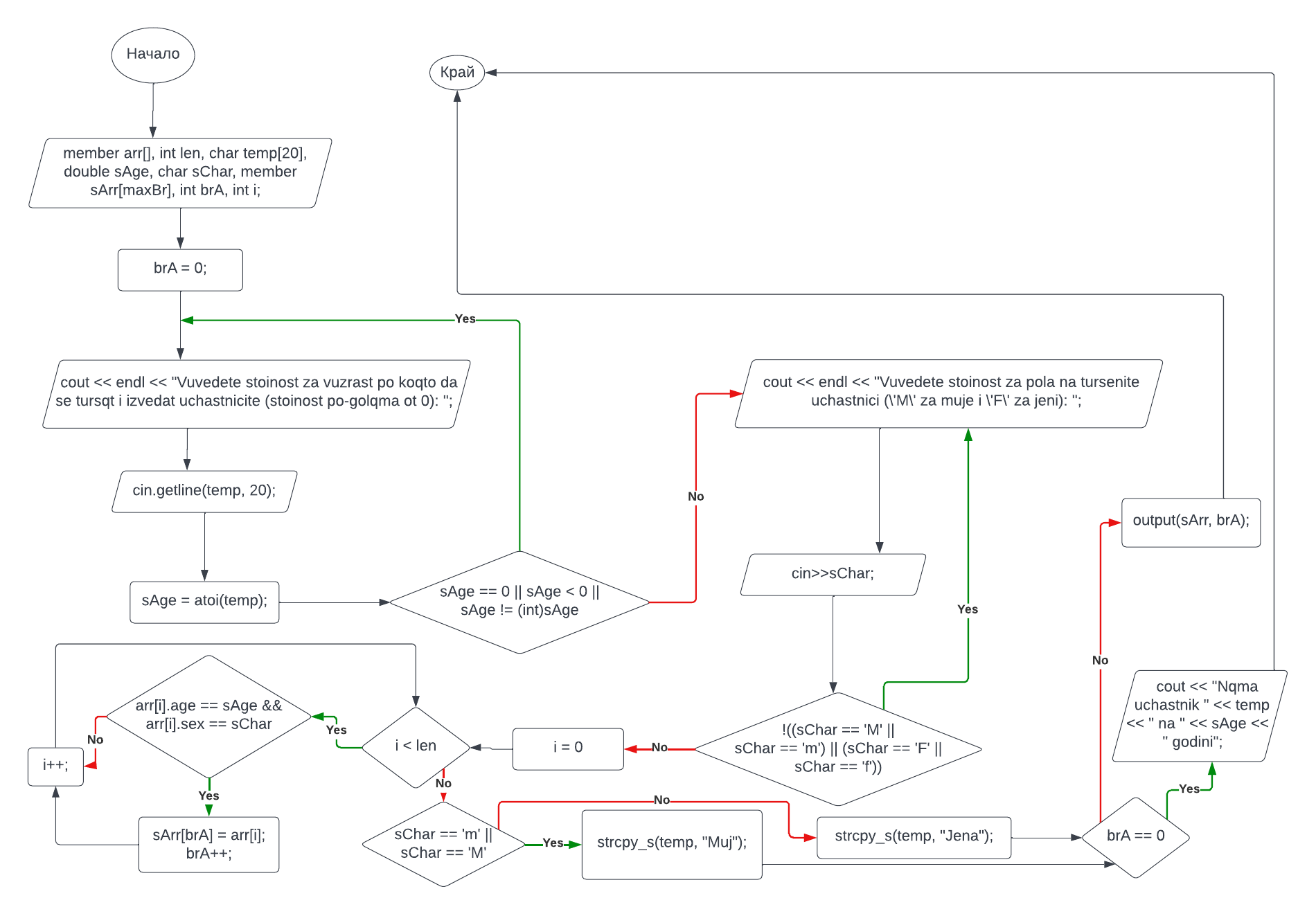
G. Въвежда се цяло число, докато не се въведе 0, при което се изпълнява функцията за меню, като въведеното число отговаря на избора на клиента от подменюто, а според избора се изпълнява една от функциите.

1. Всички участници от масива се записват в допълнителен масив, който се сортира по години, като ако срещне участници с равна възраст ги сравнява по име и ги сортира по азбучен ред, след което се извежда на екрана, като по този начин не се променя подредбата в основния масив.
2. Въвеждат се възраст и пол от клавиатурата и масивът се обхожда и ако срещне участник с възраст и пол, равни на въведените, го въвежда в допълнителен масив, който след края на цикъла се извежда.

Блок схема на алгоритъма

podmenu()

sortAgeName()

searchAgeSex()

Функция с която е реализиран алгоритъма

void podmenu(member arr[], int& len, member gr1[], member gr2[], member gr3[], member gr4[], int& len1, int& len2, int& len3, int& len4);

Функцията podmenu(), работи по същия начин като menu(), като извежда на екрана възможни избори за потребителя и чака въвеждане на число, отговарящо на избора на клиента, като при въвеждане на 0 не излиза от програмата а изпълнява функцията menu(), т.е. отваря първото меню.

void sortAgeName(member arr[], int len);

Функцията sortAgeName, въвежда данните от масива, въведен като параметър във втори масив, след което подрежда елементите във втория масив по години, като ако срещне участници с еднаква възраст ги сравнява по име и ги подрежда по азбучен ред и по този начин не се нарушава подредбата на основния масив (member arr[]).

void searchAgeSex(member arr[], int len);

Функцията searchAgeSex, търси и въвежда във втори масив, различен от този въведен като параметър всички участници с възраст и пол, равни на възраст и пол, които се въвеждат по време на изпълнение на програмата, като след това извежда данните от новия масив на екрана.

#### Входни данни на функцията

void podmenu(member arr[], int& len, member gr1[], member gr2[], member gr3[], member gr4[], int& len1, int& len2, int& len3, int& len4);

Функцията podmenu() приема като параметри масив от участници, който се използва за функциите, които се изпълняват от подменюто, също приема и брой на участниците. Останалите 8 параметъра са свързани с последната задача за разделяне на основния масив на 4 групи, като всяка една група отговаря на различен масив, но понеже от подменюто се изпълнява функцията за главното меню, в което се използват тези групи, трябва и в тази функция да се предадат 4-те групи, заедно с размерите им.

void sortAgeName(member arr[], int len);

Функцията sortAgeName() приема като параметри масива, който ще сортира и изведе и броя на елементите в него, който се използва за условие в цикъла, който се използва при сортирането на масива.

void searchAgeSex(member arr[], int len);

Функцията searchAgeSex() приема като параметри масива, в който ще се търси участник с пол и възраст равна на въведените от клавиатурата по време на изпълнение на програмата.

#### Изходни данни на функцията или данни, които се извеждат

Функцията podmenu() извежда на екрана възможните избори на клиента и не връща нищо като резултат и затова е тип void.

Функцията sortAgeName() не връща резултат и затова е тип void. Извежда на екрана подредения масив без да променя реда на основния.

Функцията searchAgeSex() не връща резултат и затова е тип void. Извежда на екрана участниците с пол и възраст равна на тези въведени от клавиатурата по време на изпълнение на програмата.

Реализация на условие … - допълнение второ

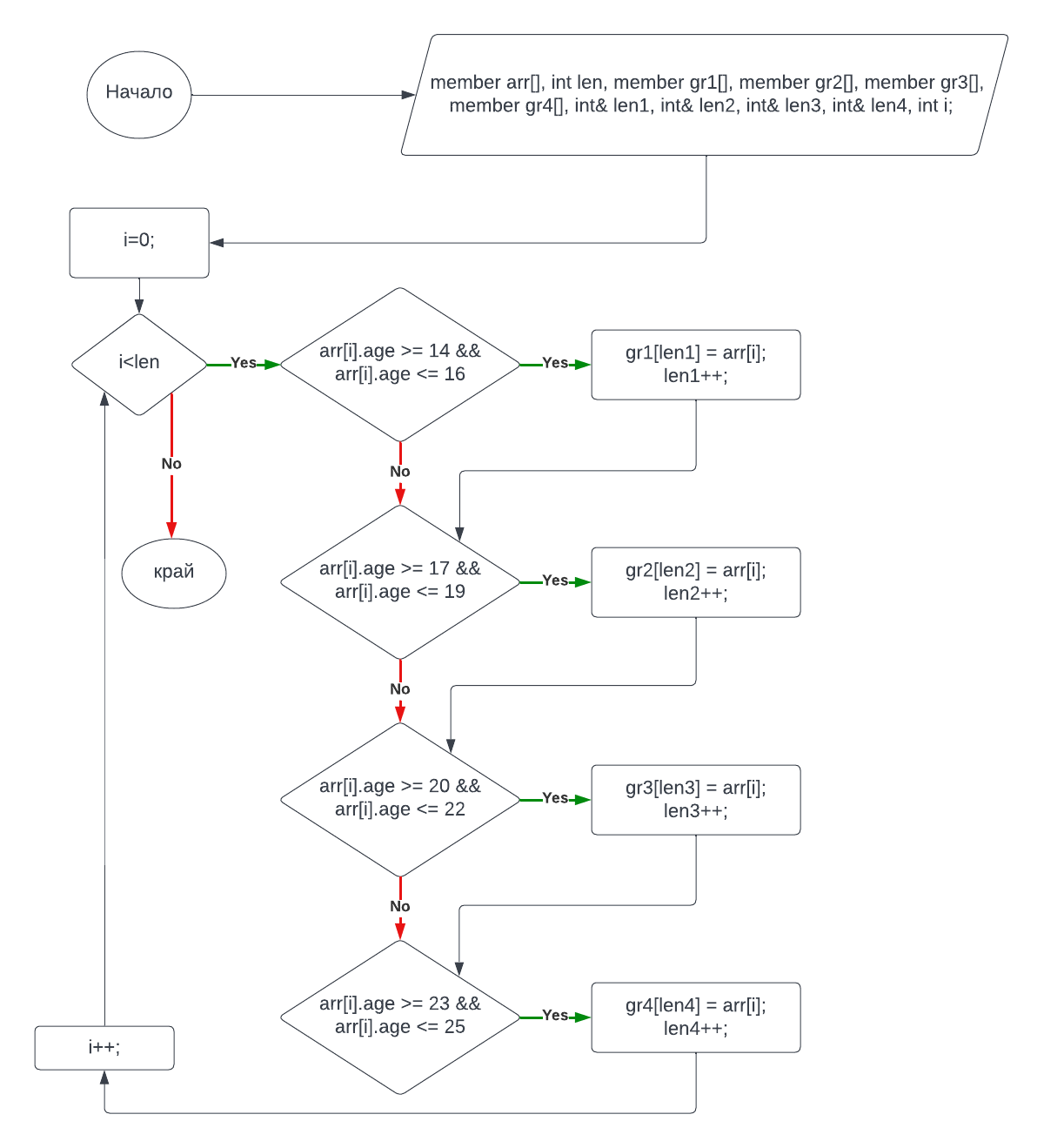
Анализ на алгоритъма, който трябва да се реализира

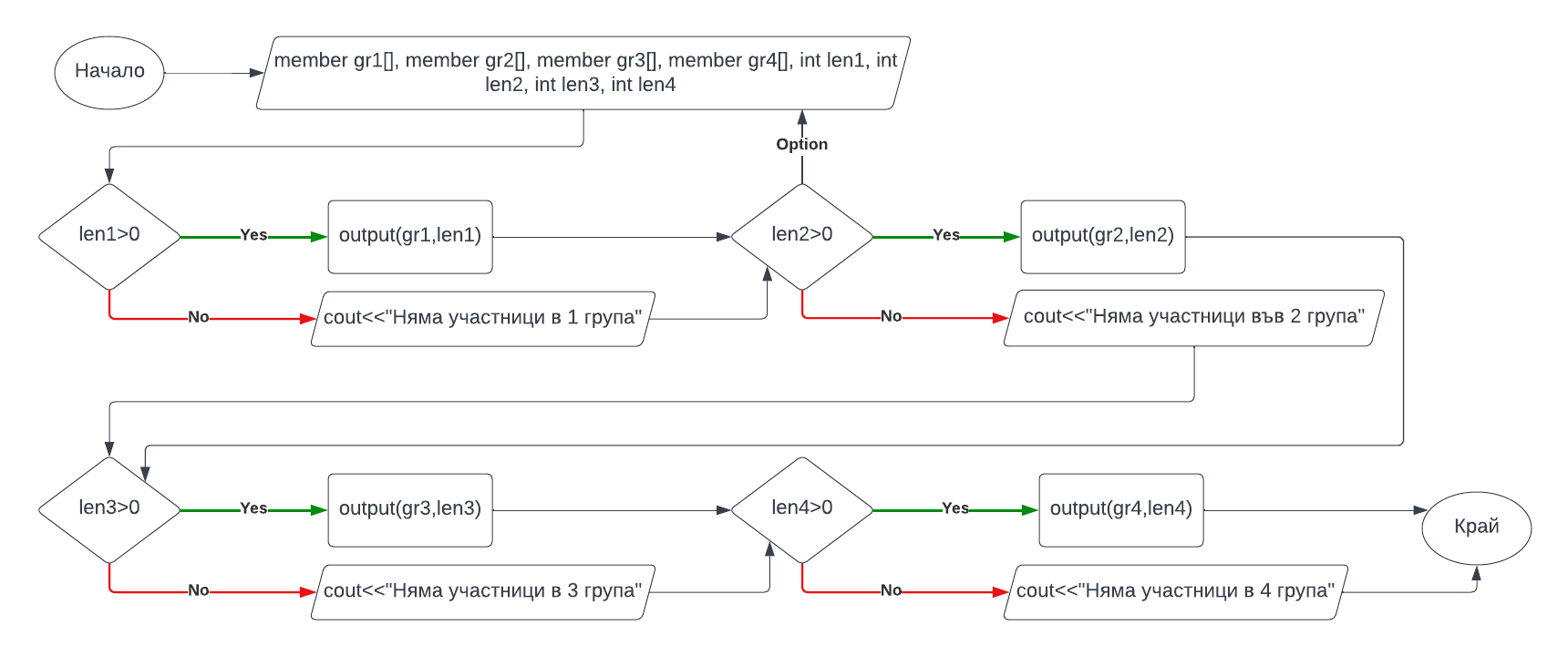
а. Обхожда се основния масив, като ако се намери участник, съвпадащ с възрастовата категория, се въвежда в допълнителен масив, отговарящ на тази категория и брояча, свързан с този масив се увеличава с 1.

b. С помощта на четирите брояча на четирите масива, отговарящи на категориите, се извеждат всички участници от всяка група и ако брояча в дадена категория е равен на 0, се извежда съобщение, че в дадената група няма участници.

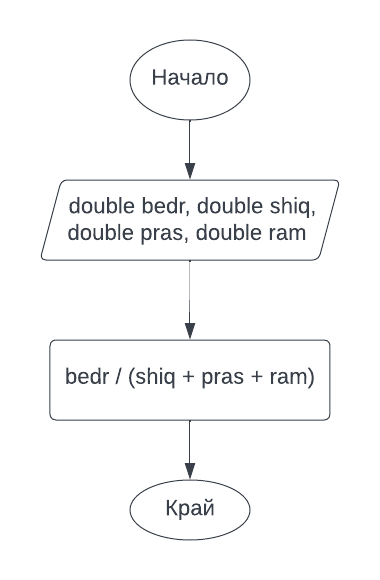
c.d.e. Обхождат се всеки един от четирите масива, като участниците в тях се разпределят в по 3 отделни масива, отговарящи на победители(П между 0,54 и 0,62), участници с П повече от 0,63 и участници с П, по-малко от 0.53, като след това всеки един от получените 12 масива се сортира и се извежда на екрана всеки един от 12-те масива, като ако няма победители в дадена група, се извежда съобщение затова.

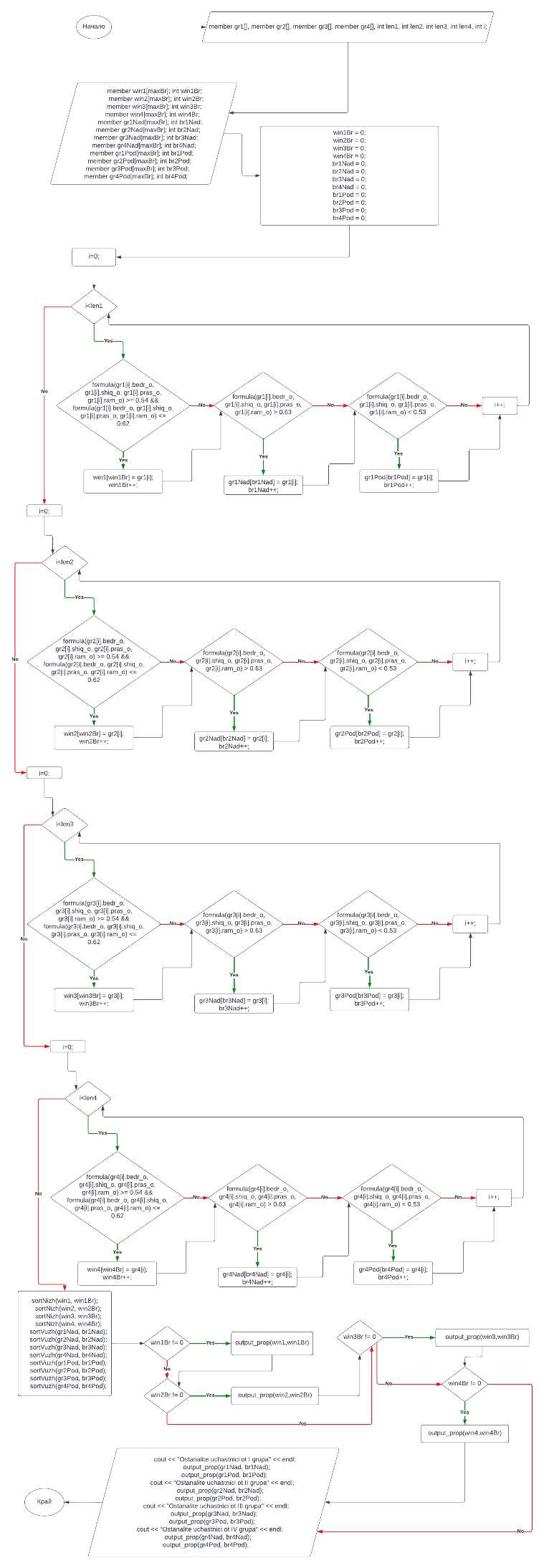
Блок схема на алгоритъма

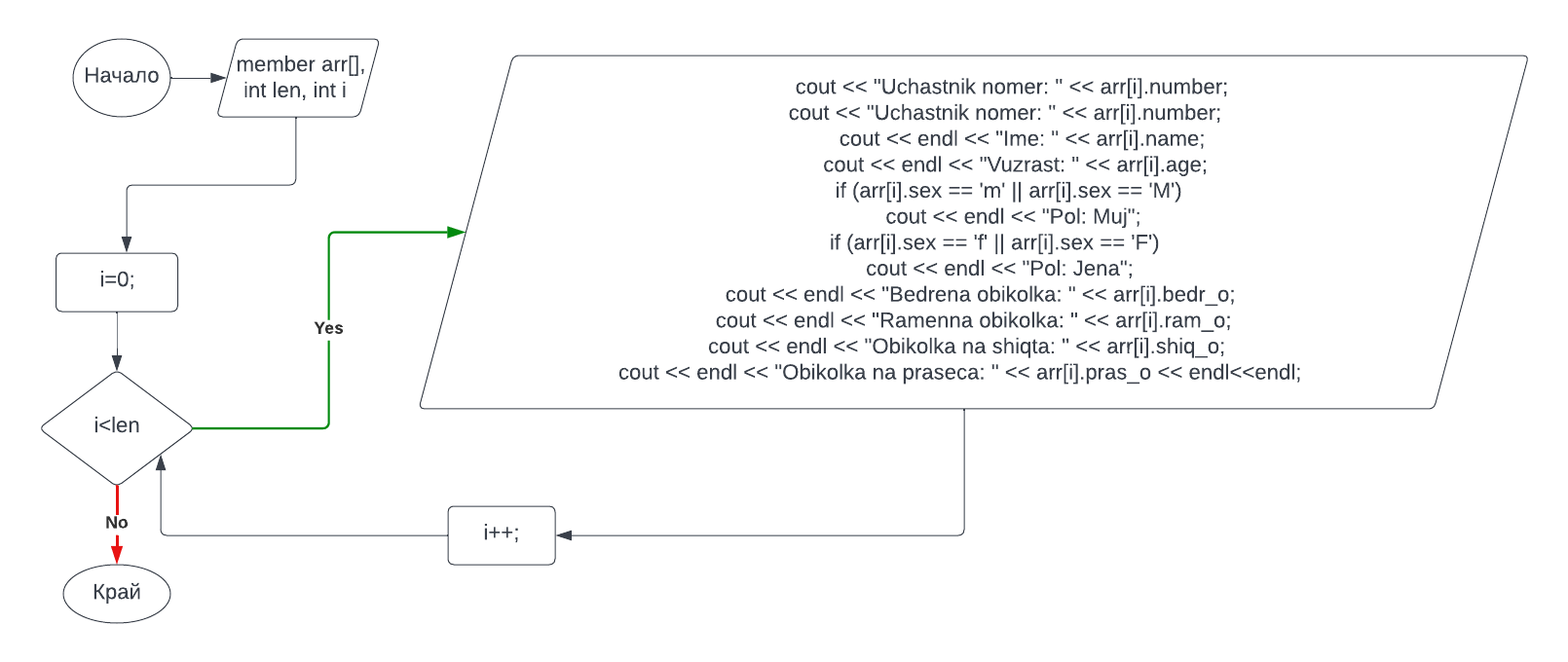
Categ()

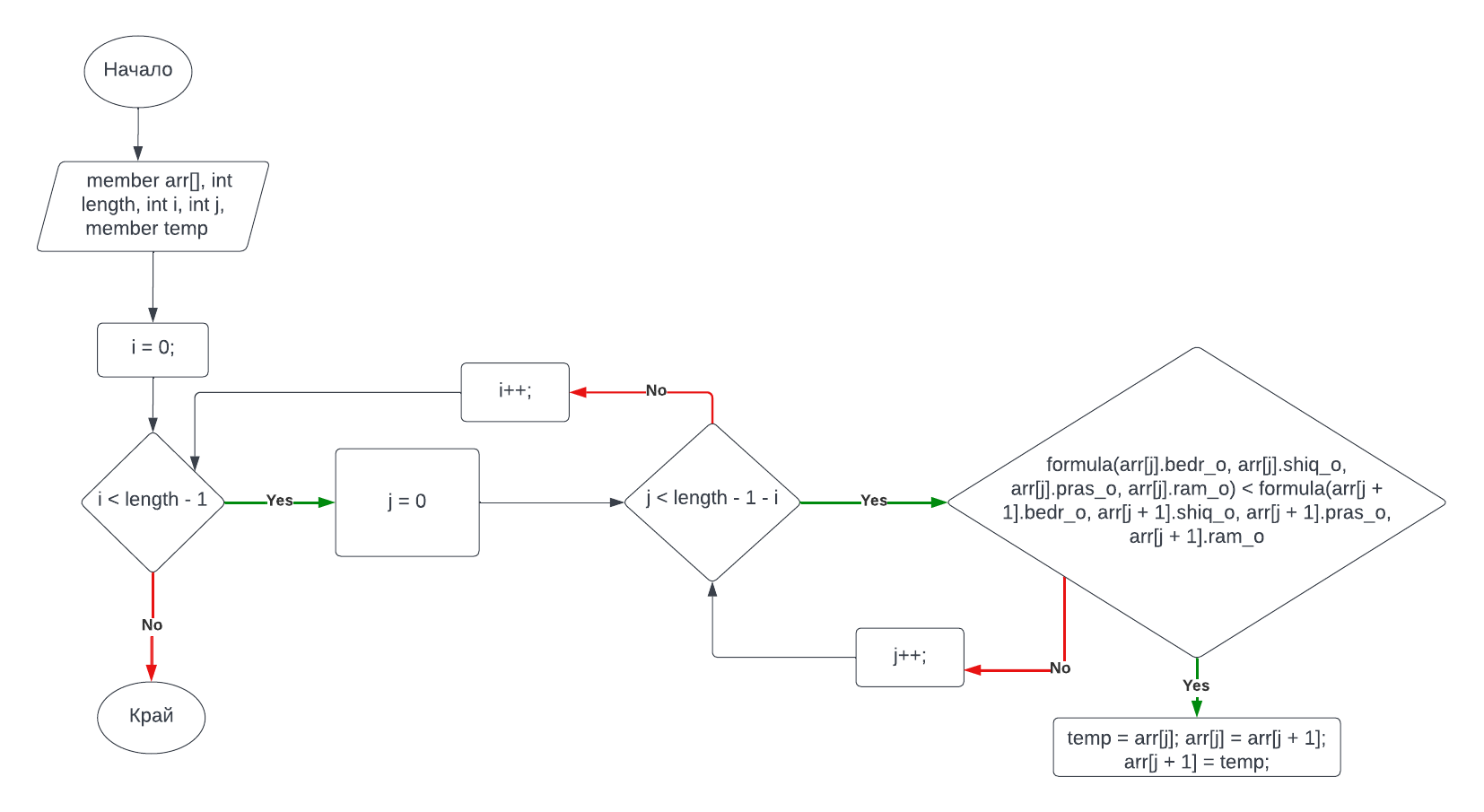
outputCat()

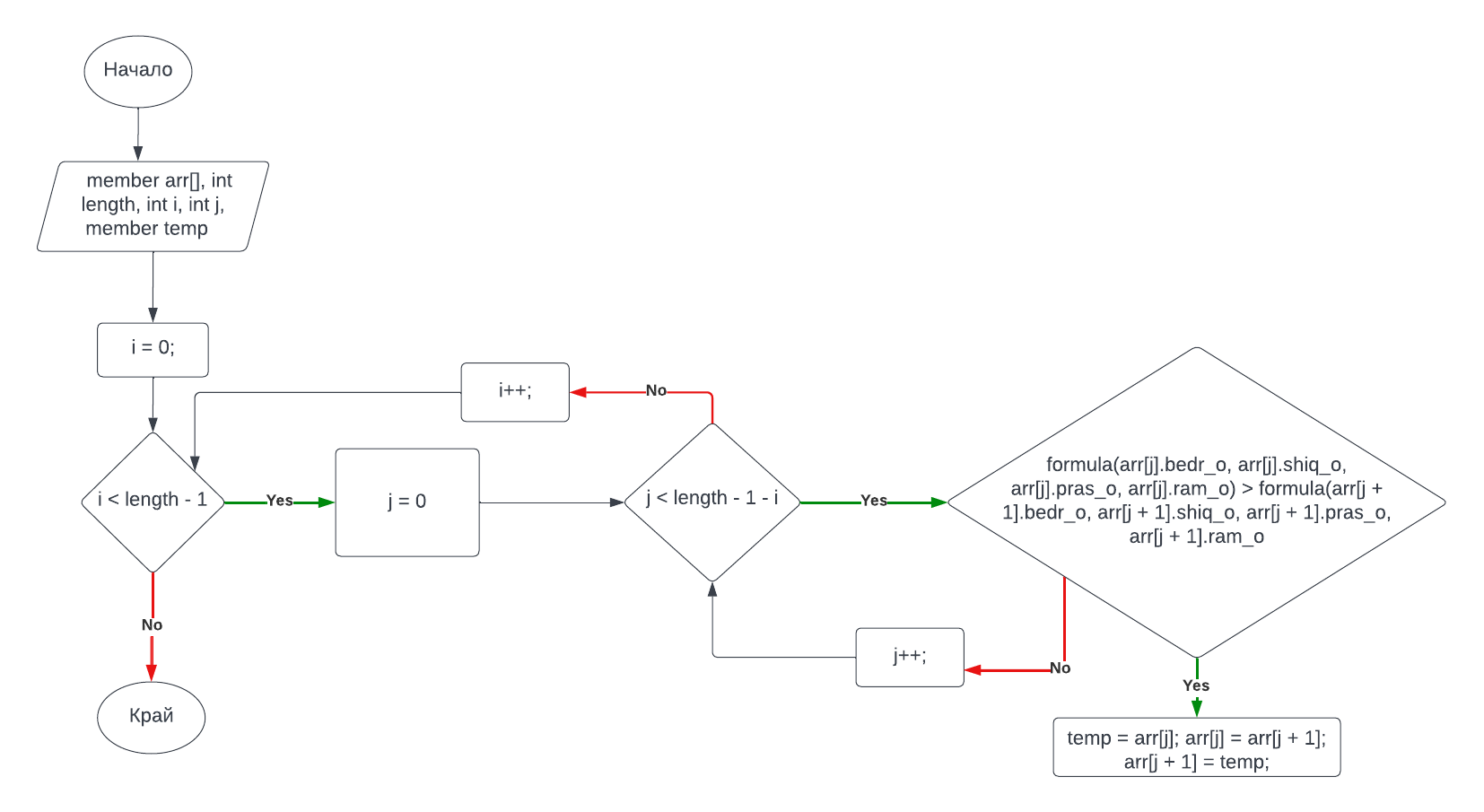
formula()



Pobeditel()

Output\_prop()

sortNizh()

sortVuzh()

Функция с която е реализиран алгоритъма

void categ(member arr[], int len, member gr1[], member gr2[], member gr3[], member gr4[], int& len1, int& len2, int& len3, int& len4);

Функцията categ() разпределя данните от основния масив member arr[], в 4 отделни масива по възрастови категории, като проверява елементите от него и в зависимост дали принадлежат на възрастовата категория, ги разпределя и увеличава броя на len1, len2, len3, len4, в зависимост от това в коя група попадне елемента, ако изобщо попадне в някоя. Тези 4 стойности отговарят на броя на елементите във всяка една от групите.

void outputCat(member gr1[], member gr2[], member gr3[], member gr4[], int len1, int len2, int len3, int len4);

Функцията outputCat() извежда на екрана всяка една от групите, а ако в дадена група няма участници, извежда съобщение затова.

double formula(double bedr, double shiq, double pras, double ram);

Функцията formula() изчислява пропорционалните телесни мерки при въведени стойности за бедрена обиколка, обиколка на шията, на прасеца и раменна обиколка.

void pobeditel(member gr1[], member gr2[], member gr3[], member gr4[], int len1, int len2, int len3, int len4);

Функцията pobeditel() разделя четирите масива, отговарящи на четирите групи на по 3 отделни масива всяка, като тези 3 масива отговарят на участници с пропорционални телесни мерки над 0.63, между 0,54 и 0,62 и телесни мерки под 0,53. След това разпределяне сортира всички 12 масива в зависимост от условието и накрая извежда данните от всеки един от тези 12 масива, като ако няма победители в групите (няма участници с пропорционални телесни мерки между 0.54 и 0.62), се извежда съобщение затова.

void output\_prop(member arr[], int len);

Функцията output\_prop() изпълнява същото действие като функцията output(), с разликата че извежда и стойността за пропорционални телесни мерки на всеки участник.

void sortNizh(member arr[], int length);

Функцията sortNizh() сортира по низходящ ред даден масив от участници member arr[], с размер int length, спрямо стойността за пропорционални телесни мерки, която се определя с функцията formula().

void sortVuzh(member arr[], int length);

Функцията sortNizh() сортира по възходящ ред даден масив от участници member arr[], с размер int length, спрямо стойността за пропорционални телесни мерки, която се определя с функцията formula().

#### Входни данни на функцията

Функцията categ() приема като входни параметри масив от участници member arr[], от който ще се категоризират участниците, брой на участниците в този масив int len, който служи за край на цикъла, с който се проверява дали даден участник отговаря на критерия на някоя от групите, 4 масива отговарящи на четирите групи и 4 стойности за брой на участниците във всяка една от групите, като този брой се променя с всеки нов участник попаднал в дадена група и затова се предават по адрес.

Функцията outputCat() приема като параметри 4 масива отговарящи на четирите групи и 4 стойност за броя на елементите във всяка една от групите.

Функцията formula() приема стойности за бедрена обиколка, обиколка на шията, на прасеца и раменна обиколка.

Функцията pobeditel() приема като стойност четирите масива, отговарящи на четирите групи, от които ще вземе данните и ще ги разпредели в 12 различни масива и броя на участниците във всяка една от групите.

Функцията output\_prop() масив от участници (member arr[]) и брой на елементите в него (int len). От масива ще се извеждат всички данни, а броя на елементите се използва за край на цикъла, който ще се върти за да се изведе всеки един участник, тоест цикъла ще продължава докато брояча не стане равен на броя на елементите (len).

Функциите sortNizh() и sortVuzh() приемат едни и същи данни – масив от участници и негова дължина или брой на елементите в него.

#### Изходни данни на функцията или данни, които се извеждат

Функциите sortNizh(), sortVuzh() и categ() не връщат резултат и не извеждат нищо на екрана и затова са от тип void.

Функцията outputCat() не връща резултат и затова е тип void. След изпълнение, ще изведе на екрана всички групи, на които е разделен масива, като ако няма участници в дадена група, ще се изведе съобщение затова, както и ако масива не е категоризиран предварително с функцията categ(), отново ще изведе съобщение, че в групите няма участници.

Функцията pobeditel() не връща резултат и затова е тип void. На екрана ще се изведат всички групи, като първо ще се изведат победителите от всяка група (участниците с пропорционални телесни мерки между 0.54 и 0.62), подредени в намаляващ ред, като ако няма победители в дадена група, ще се изведе съобщение затова. След това, за всяка група ще се изведат останалите участници, като първо ще се изведат тези с П повече от 0.63 в нарастващ ред, след които следват участниците с П, по-малко от 0.53, които отново са наредени в нарастващ ред.

Функцията formula() връща резултат реално число, отговарящо на стойността на пропорционални телесни мерки на даден участник, и затова е тип double.

Реализация на допълнение трето

Анализ на алгоритъма, който трябва да се реализира

Моля, опишете.

При отваряне на файла, преди изпълнението на функцията за меню се изпълнява функцията за четене, която определя броя на участниците и записва данните от файла в основния масив.

При затваряне на файла или при въвеждане на 0 от клавиатурата в менюто, преди функцията exit(0), се изпълнява функцията за запис, като по този начин автоматично при затварянето на програмата се записват данните от масива в двоичния файл.

Блок схема на алгоритъма

Тук се поставя блок схема на описания алгоритъм.

Функция с която е реализиран алгоритъма

void zapis(member arr[], int len);

void chetene(member arr[], int &len);

За реализацията на допълнение трето съм използвал същите функции, с които записвам и чета данните от файл, в условие F.

При отварянето на програмата, преди изпълнението на функцията за менюто, се изпълнява функцията за четене на данните от файл. Тя записва всичко от бинарния файл и го слага в паметта на компютъра в масива member arr[], след което в променливата int& len записва броя на участниците от масива.

При затваряне на програмата или при избор на 0 от основното меню, преди да се затвори програмата, данните се записват в двоичния файл.

#### Входни данни на функцията

И двете функции приемат като параметър масив от участници (member arr[]), където ще се записват данни, или откъдето ще се четат данни, които после да се въведат в двоичен файл. И двете функции приемат като параметър размер на масива или броя на участниците в него (int len или int& len) като тази стойност се използва, да се каже на програмата колко пъти да прочете размера на участника от паметта или от файла. При четене тази стойност се намира като с функцията seekg() се отиде в края на файла и с tellg() се определи броя на байтовете във файла, след което този брой се раздели на размера на размера на участника member, като така се определя броя на елементите, записани във файла.

#### Изходни данни на функцията или данни, които се извеждат

И двете функции не връщат резултат и не извеждат нищо на екрана. Те записват данни от масив във файл или четат данни от файл и ги записват в масив от паметта.

Упътване за употреба

Очаквани входни данни

A. Функцията menu() очаква цяло число като входни данни в интервала [0;11], като при въвеждане на число, извън този интервал се изисква повторно въвеждане.

B. a. Функцията addd(), чака въвеждането на цяло число, което е по-малко от броя участници вече записани в масива, след което ако числото отговаря на критерия започва въвеждане на толкова участници, като за целта от клавиатурата се въвеждат всички данни за участниците. При грешно въведено число за брой на участниците, които ще се въвеждат се извиква повторно въвеждане на числото, до момента в който не се въведе число, което отговаря на критерия. При въвеждането на участниците се въвежда цяло число за номера на участника, по-голямо от 0, име на участника, до 50 символа, в което могат да присъстват и разделители, освен \n, възраст, която отново трябва да е цяло число, по-голямо от 0, пол, който приема стойност символите m М за мъжки пол и f F, за женски, стойности за бедрена и раменна обиколка, както и обиколка на шията и прасеца, които могат да бъдат реални числа. При въвеждане на символ или дума, се изисква повторно въвеждане на съответните данни на участника

C. a. Функцията output(), не изисква входни данни.

D. a. Функцията minAge не изисква входни данни от клавиатурата

b. Функцията searchName() очаква като входни данни име, което да търси в масива с участници.

E. a. Функцията sortAge() не изисква входни данни от клавиатурата.

F. a.b. Функциите zapis() и chetene() не изискват входни данни от клавиатурата.

G. Функцията podmenu() чака въвеждането на цяло число, отговарящо на номера на опция от менюто в интервала [0;2], като при въвеждане на число, извън този интервал се изисква повторно въвеждане.

а. Функцията sortAgeName() не изисква входни данни от клавиатурата.

b. Функцията searchAgeSex() очаква въвеждането на възраст, цяло число по-голямо от 0 и пол,m M или f F, като при грешно въвеждане на данните се изисква повторното им въвеждане.

H.

а. Функцията categ() не изисква входни данни от клавиатурата.

b. Функцията outputCat() не изисква входни данни от клавиатурата.

I. Функцията pobeditel() не изисква входни данни от клавиатурата

c. Функцията output\_prop() не изисква входни данни от клавиатурата

Очакван резултат от изпълнението на конкретния фрагмент от проекта

При въвеждане на цялото число 1 от клавиатурата, при изпълнение на функцията menu(), ще се изпълни функцията addd() и след въвеждане на числото 1 ще започне въвеждането на нов участник, ако в масива има свободни места. Ще се чака въвеждане на поредния му номер, името и т.н., след което тези данни ще се запишат в масива и програмата ще изпълни функцията menu().

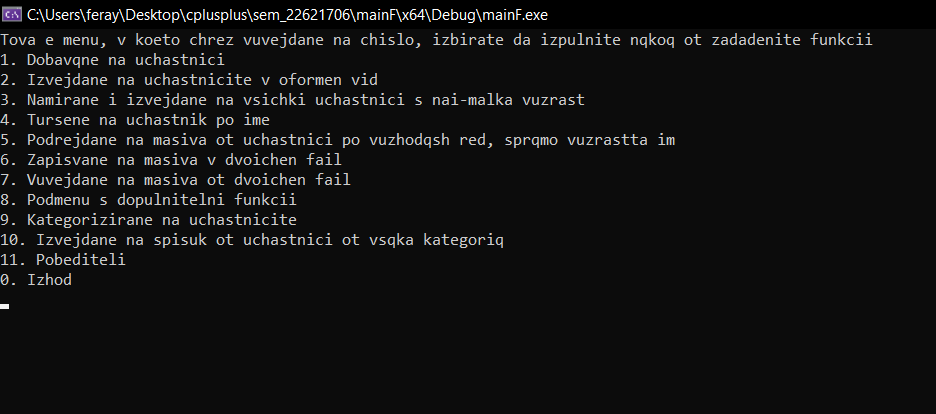
Очакван резултат от изпълнението на конкретния фрагмент от проекта

При изпълнението на функцията podmenu(), след въвеждане на цялото число 2, ще се изпълни функцията searchAgeSex() и след въвеждане на 14 m, ще търси участници с пол m или M и възраст 14 и ако намери такива, ще ги изведе на екрана.

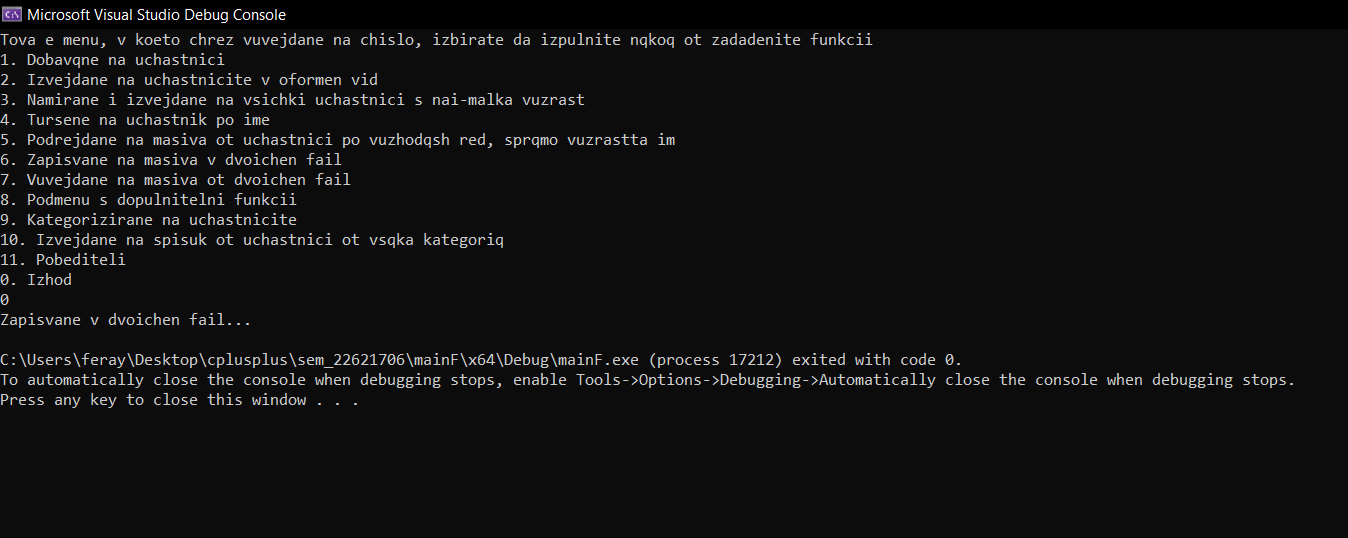
Примерно действие на програмата

Условие A

Снимка на изгледа с примерни входни данни

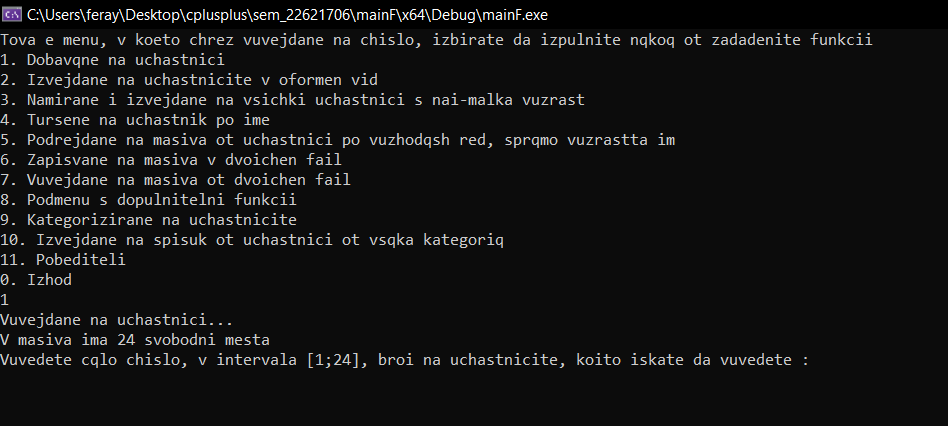


Снимка на изгледа с примерни изходни данни

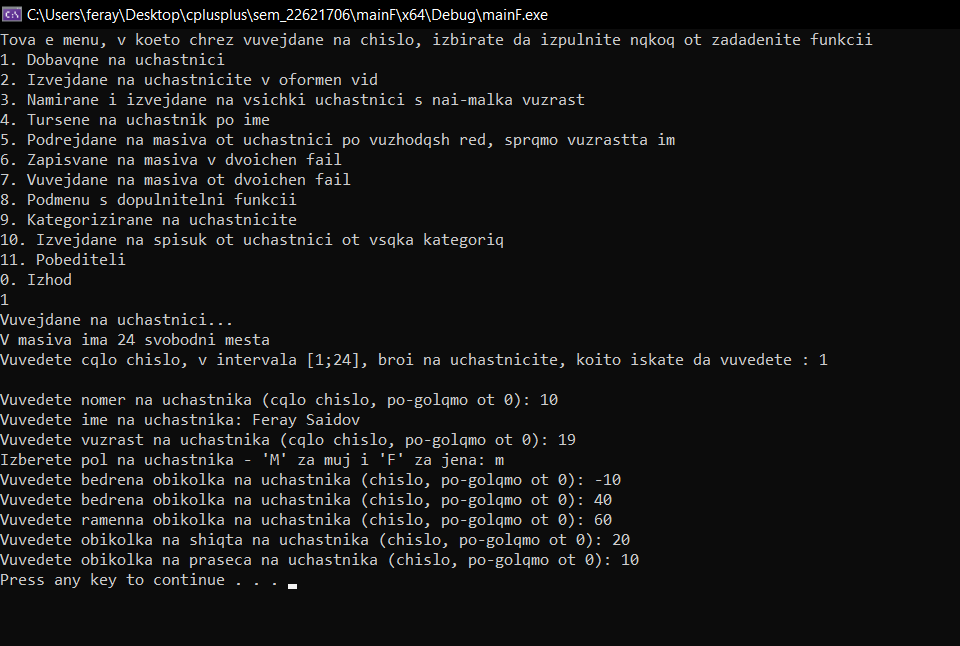


Условие B

Снимка на изгледа с примерни входни данни



Снимка на изгледа с примерни изходни данни



Условие C

Снимка на изгледа с примерни входни данни

Снимка на изгледа с примерни изходни данни

Text

Description automatically generated

Условие D

Снимка на изгледа с примерни входни данни

Снимка на изгледа с примерни изходни данни

Text

Description automatically generated

Text

Description automatically generated

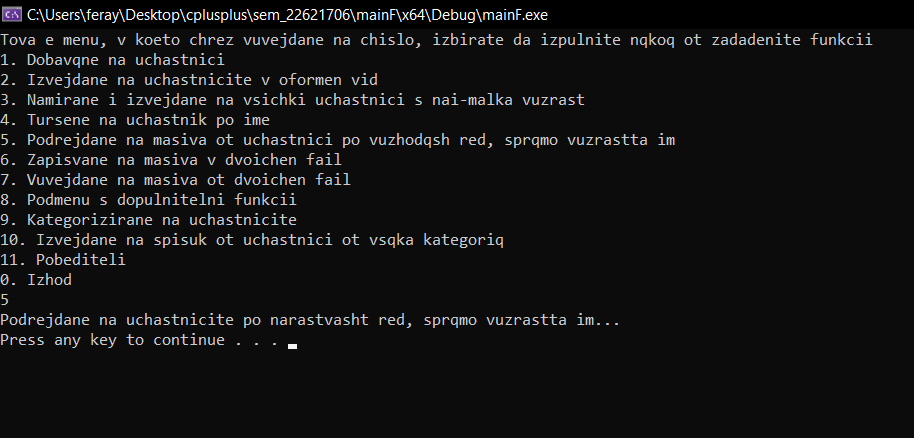
Text

Description automatically generated

Условие E

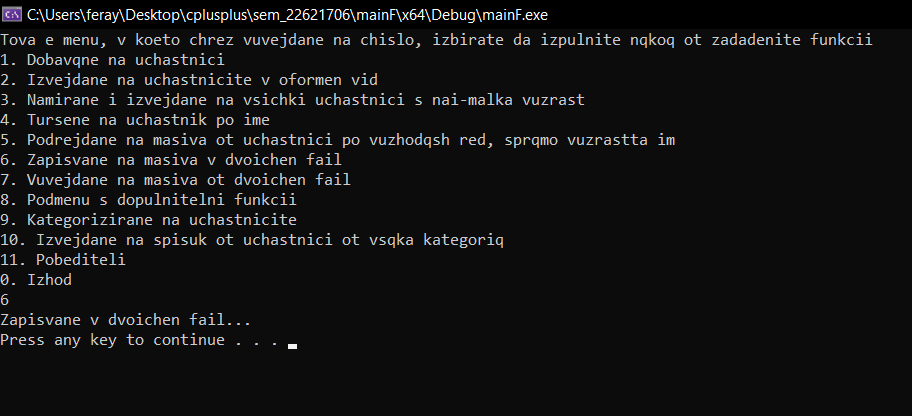
Снимка на изгледа с примерни входни данни

Снимка на изгледа с примерни изходни данни

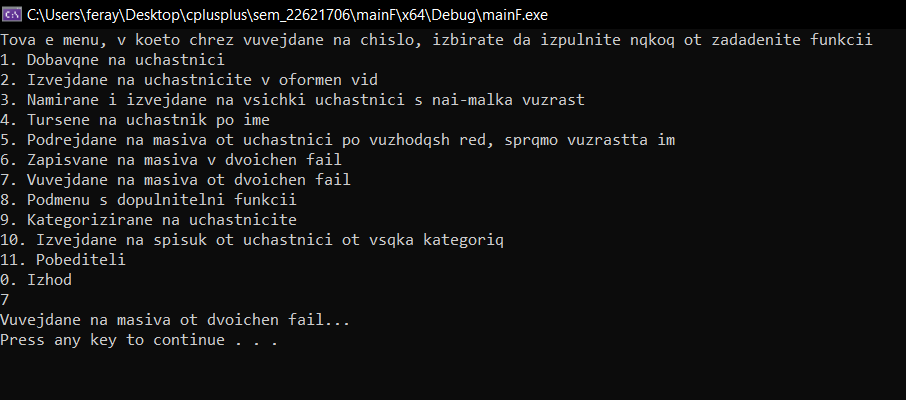


Условие F

Снимка на изгледа с примерни входни данни

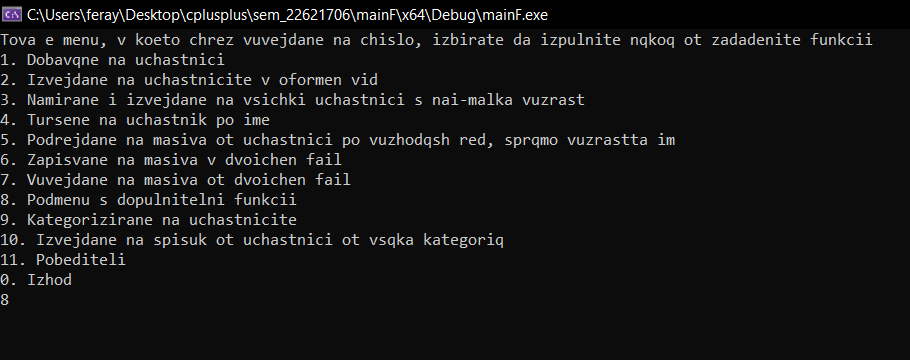


Снимка на изгледа с примерни изходни данни

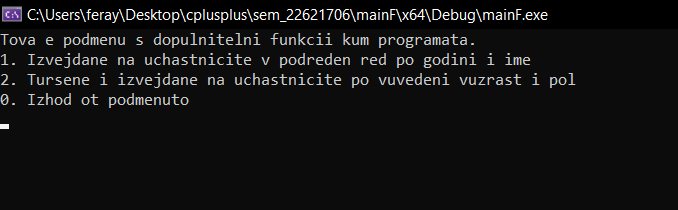


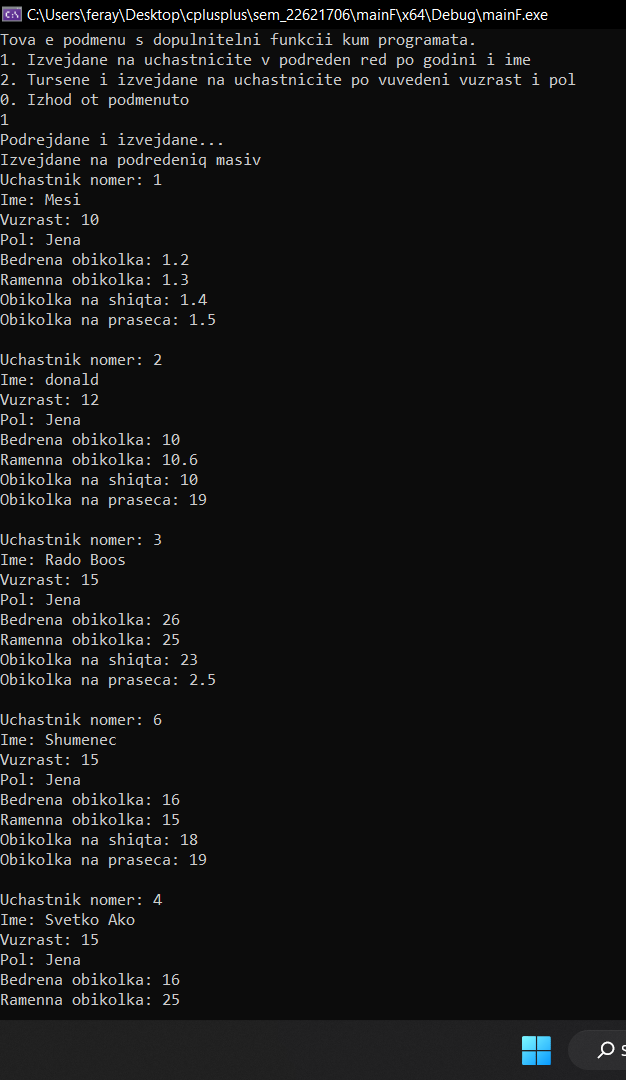
Допълнение първо

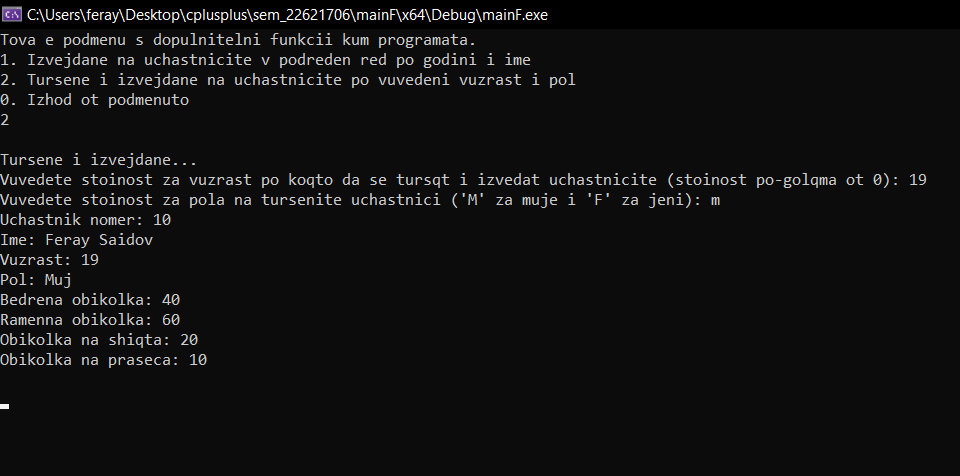
Снимка на изгледа с примерни входни данни

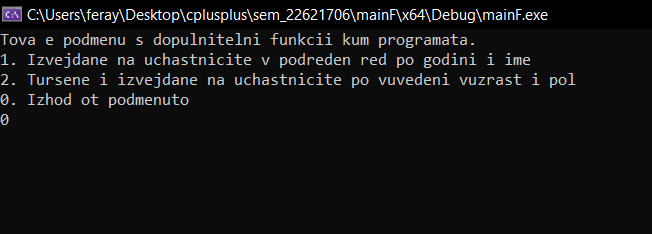


Снимка на изгледа с примерни изходни данни







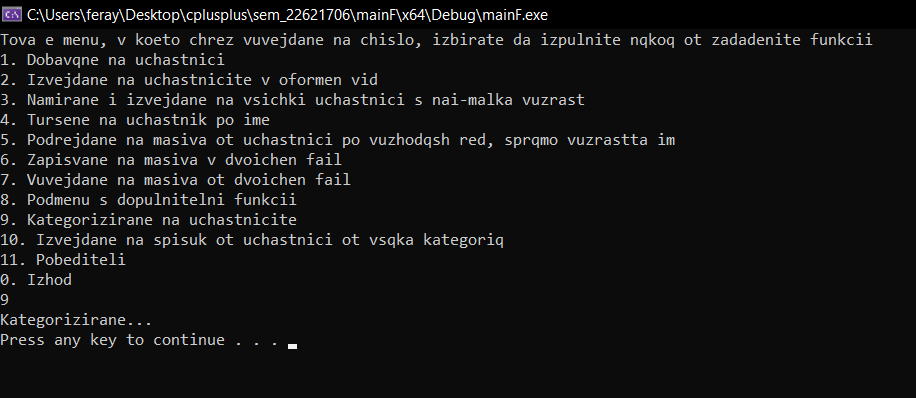


Text

Description automatically generated

Допълнение второ

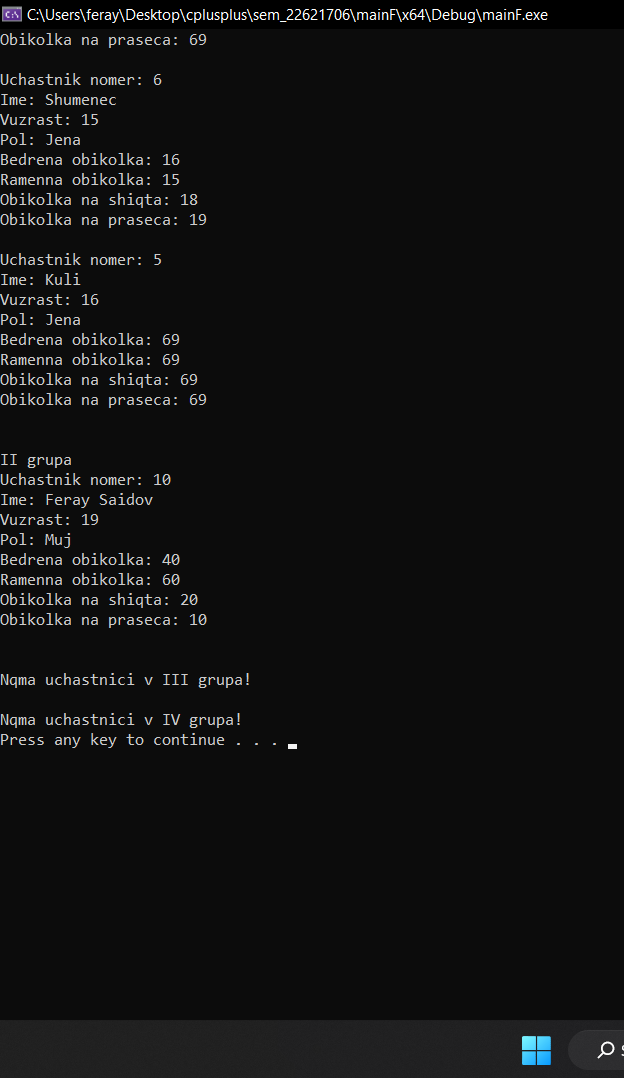
Снимка на изгледа с примерни входни данни



Снимка на изгледа с примерни изходни данни

Text

Description automatically generated



Допълнение трето

Снимка на изгледа с примерни входни данни

Text

Description automatically generated

Снимка на изгледа с примерни изходни данни

