**ТЕХНИЧЕСКИ УНИВЕРСИТЕТ ВАРНА**

**Факултет по изчислителна техника и автоматизация**

**Софтуерни и интернет технологии**

**семестриална домашна работа: факултет**

**вариант 87**

**Изготвил: Златин Петров Великож**

**20691582**

Вариант | **87**

20621582 ЗЛАТИН ПЕТРОВ ВЕЛИКОВ

# Факултет

Да се напише компютърна програма, реализираща информационна система, която поддържа специалности във факултет. Програмата съхранява и обработва данни за студентите в специалности (КСТ и СИТ, факултетен номер, Име, година и месец на раждане, Списък с дисциплини). Максималния брой студенти от двете специалности е 30. Всеки студент има 5 оценки по дисциплините (Име, оценка) от 1-ви семестър.

## Базова задача – Сложност ниска

1. Меню за избор на функциите в програмата. (7 седмица)
2. Добавяне на студенти в група (7-8 седмица)
   1. Добавяне на един студент в групата с оценки по 5 дисциплини от 1 семестър;
   2. Добавяне на списък от студенти. Въвежда се цяло число n и след него n на брой студенти с оценки по 5 дисциплини от 1 семестър.
3. Извеждане на всички студенти на екрана (8-9 седмица)
   1. Извеждане на студенти от дадена специалност
   2. Извеждане на студенти с оценка Отличен по БП

## Допълнение Първо – Сложност средна (+ Базова задача)

D. Разпределение на студентите: (9-10 седмица)

1. Една група се състои от максимум 5 студента и минимум 4 от една специалност, разделете студентите в различни масиви за всяка;
2. Ако след разпределението има студенти без групи те се разпределят по 1 във вече сформираните групи.
3. Ако има трима останали студенти могат да се сформират две групи по 4-ма студенти.

## Допълнение Второ – Сложност висока (+ Базова задача + Допълнение Първо)

E. Промяна на студентите в групата (10-11 седмица)

1. Въвеждат се факултетните номера на двама студенти и се разменят групите им.
2. Въвежда се факултетен номер на студент за да се изтрие от група, ако група остане с по малко от 4 студенти се разпределят по правилата в т.(D);

## Допълнение Трето – Сложност висока (+ Базова задача + Допълнение Първо + Допълнение Второ)

1. Справки за студентите във факултета с под меню (11-12 седмица)
   1. Извеждане на студентите в подреден ред по:
      1. Среден успех;
      2. По възраст в месеци на студент.
   2. Извеждане на студентите с най-добър среден успех от КСТ и СИТ;
   3. Извеждане на студентите с успех над 5,50.
2. Данните в програмата да могат да се запазват във файл между две стартирания на програмата.

## Допълнение Четвърто – (за допълнителни точки)

H. Допълнителни условия:

1. За точка B при добавянето на един студент да се даде възможност, да се избере дали да се въведе нов или да прекъсне въвеждането.
2. За точка C да се реализира изписването по 5 студента на страница. Пример: Във факултета има 30 студента. Извеждат се 5 студента, избира се следваща страница извеждат се следващите 5 студента и т.н.
3. За точка G да се записват данните в двоичен файл.

**Трябва да се добавят тези файлове за да работи.**

За изпълнението на

* Първа част са използвани - struct student, int main, void input, void find\_high\_grade, void find\_specialization
* Втора част са използвани - struct student, struct group, int main, void group\_distribution
* Трета част са използвани - struct student, struct group, int main, void group\_distribution, void add\_record, void delete\_record, void change\_record\_menu
* Четвърта част са използвани - struct student, int main, void report\_menu, void sort\_by\_gpa, void sort\_by\_age, void find\_high\_GPA, void greatest\_GPA

struct grade\_full{

string grade\_name; -име на оценка

double grade [5]; -входни оценки по даден предмет

double finale\_grade; -срочни по даден предмет

};

struct student{

string full\_name ; -име на ученик

string specialization ; -име на специалност

unsigned int fact\_nomer; -факултетен номер на учееник

int month\_of\_birth; -месец на раждане

int year\_of\_birth; -година на раждане

int age\_month; -възраст на ученик в месеци

grade\_full semester [5]; -предметите

double finale\_grade\_semester; -семестрялна оценка на ученик

};

struct group{

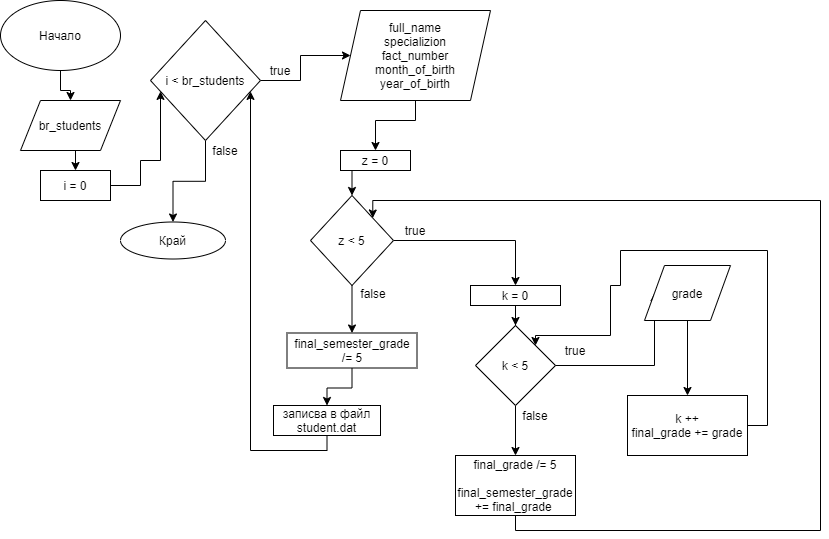
student group\_students [5]; -учениците в групата

string specialization; -специалност на групата

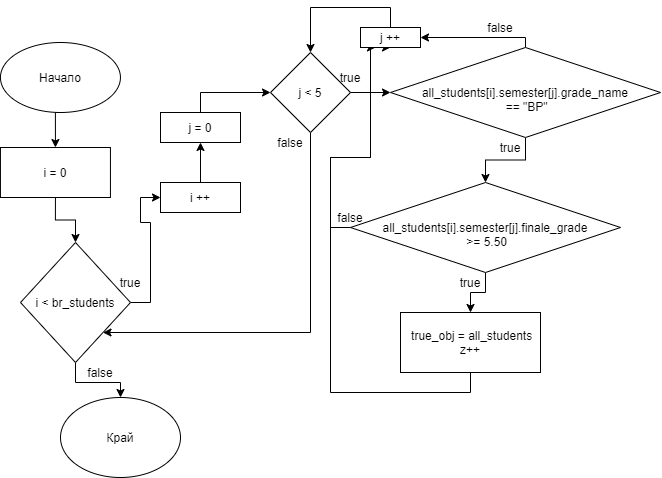
int br\_group\_students; -брой ученици в групата

};

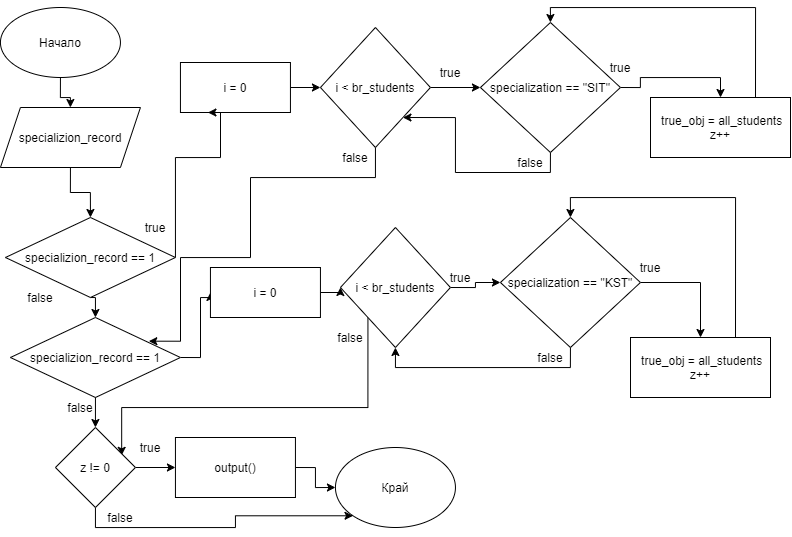
void input(student\* all\_students, int& br\_students) – функцията служи за първоначално въвеждане на няколко ученици, в нея се въвежда n на брой студенти, след това се въвежда информацията за ученика и се пресмятат оченките, накрая се записват в фаил./SIT или KST; първото име на англиски; факултетен номер; месец и година на раждане и оченки/

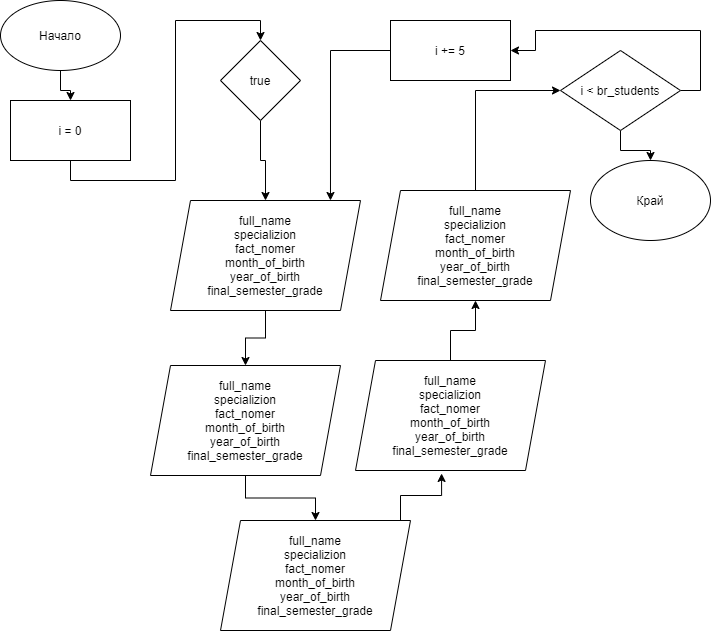


void find\_high\_grade(student all\_students[], int br\_students)- намира всички ученици с 5,50 и нагоре по БП/вфодни няма/

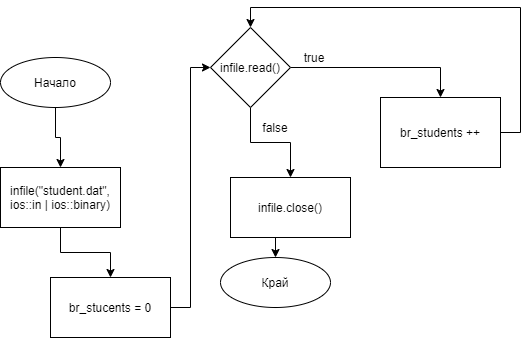


void find\_specialization(student all\_students[], int br\_students)- по въведена специалност, извежда учениците по тази специалност/1 или 2/

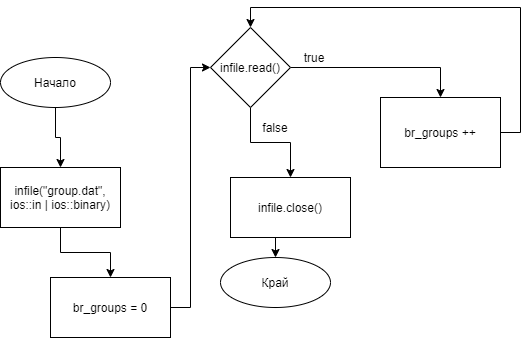


void output(student all\_students[], int br\_students)- изписва страници с учениците/няма входни данни/

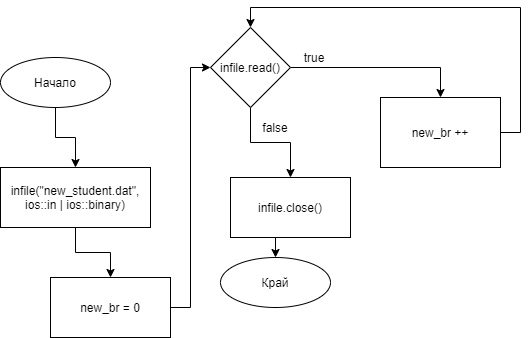
void read\_file(student\* all\_students, int& br\_students)-чете учениците от файла



void read\_file\_group(group\* all\_students, int& br\_group, bool& flag\_group)-чете групите от файла

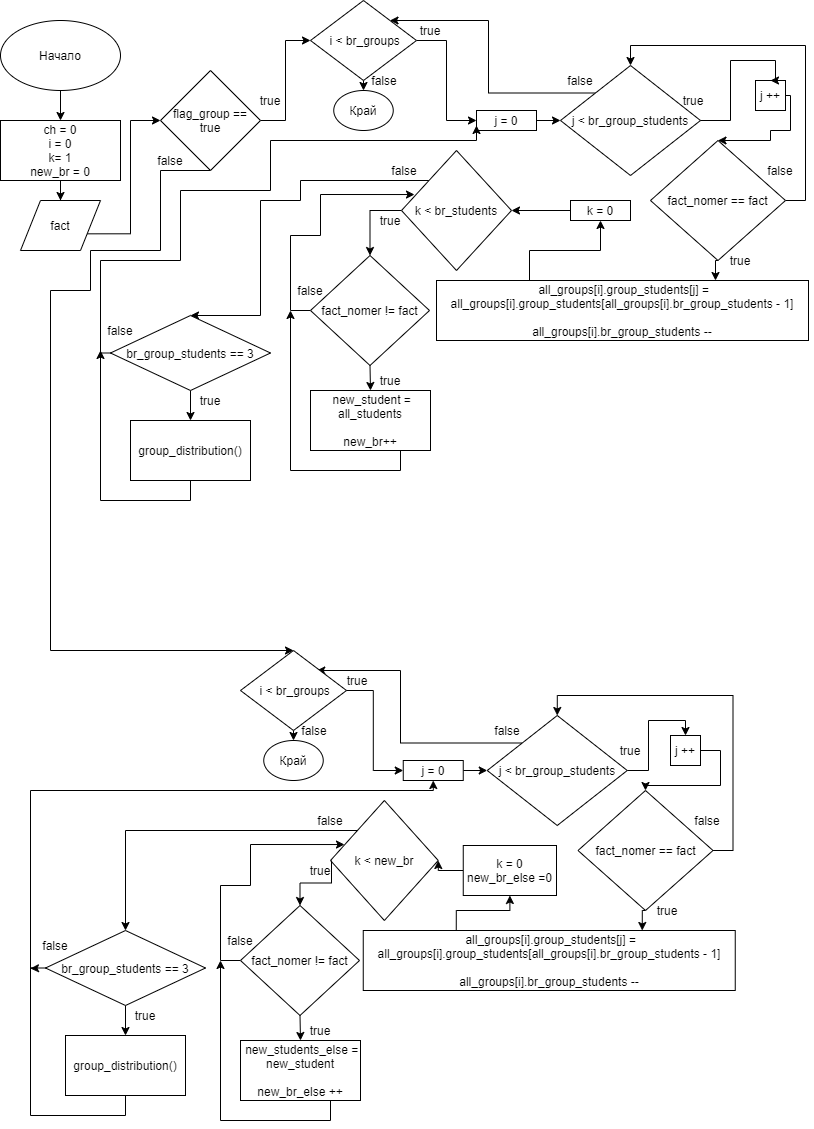


void read\_file\_new(student\* new\_students, int& new\_br) – чете студенти само когато има изтрити



void add\_record(student\* all\_students, int& br\_students) – функцията служи за добавяне на ученик след началните, след това се въвежда информацията за ученика и се пресмятат оченките, накрая се записват в фаил /SIT или KST; първото име на англиски; факултетен номер; месец и година на раждане и оченки/.

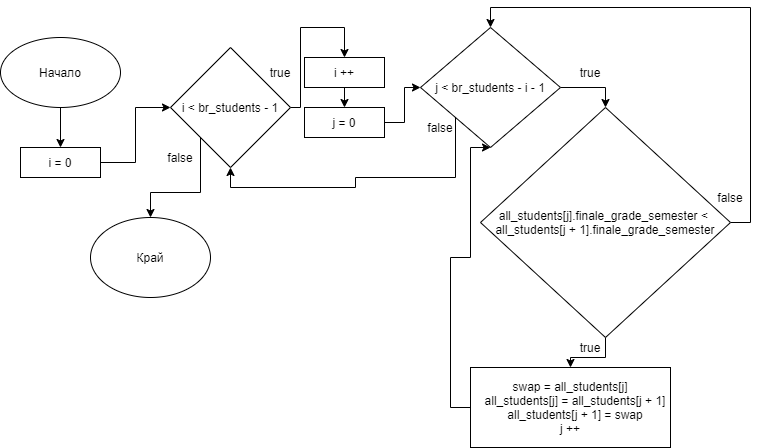


void delete\_record(group\* all\_groups, int& br\_groups, student\* all\_students, int& br\_students, bool& flag\_group)- трие ученик от група по зададен факултетен номер/ факултетен номер/

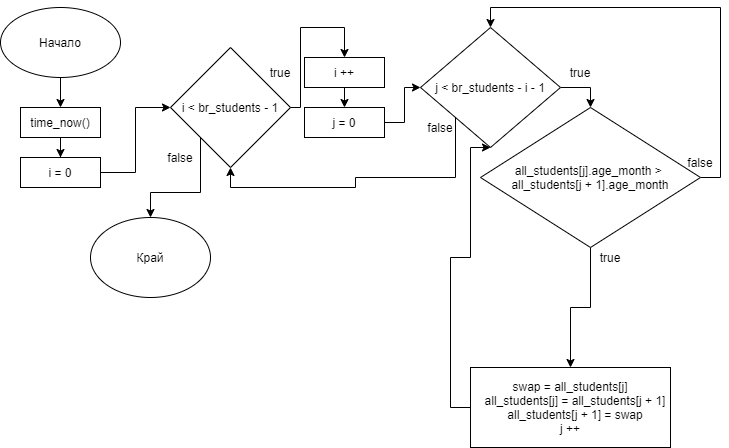
void change\_record\_menu(student\* all\_students, int& br\_students, group\* all\_groups, int& br\_groups, bool& flag\_group) -меню за добавяне или изтриване на ученик

void report\_menu(student\* all\_students, int& br\_students, int& year, int& month) -меню за справки по ученици от **Допълнение Трето**

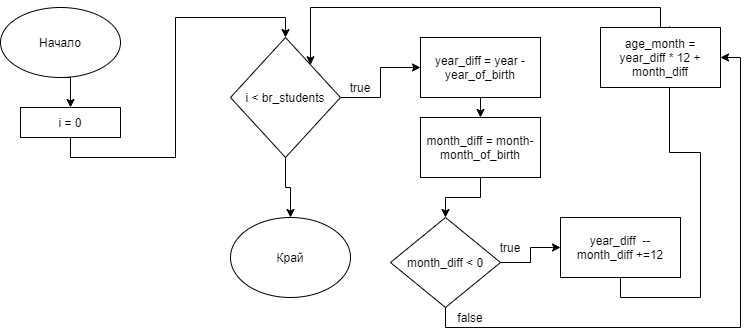
void sort\_by\_gpa(student\* all\_students, int& br\_students) – сортира ученици по успех



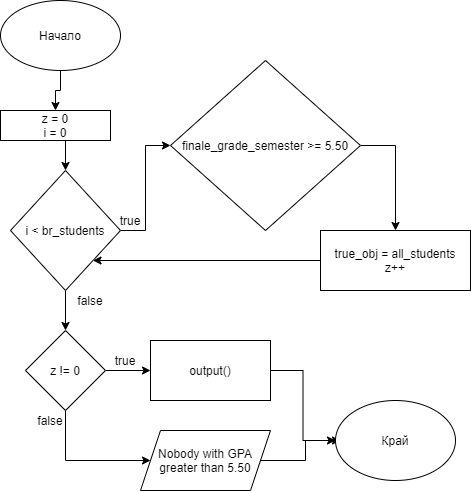
void sort\_by\_age(student\* all\_students, int& br\_students, int& year, int& month) - сортира ученици по възраст



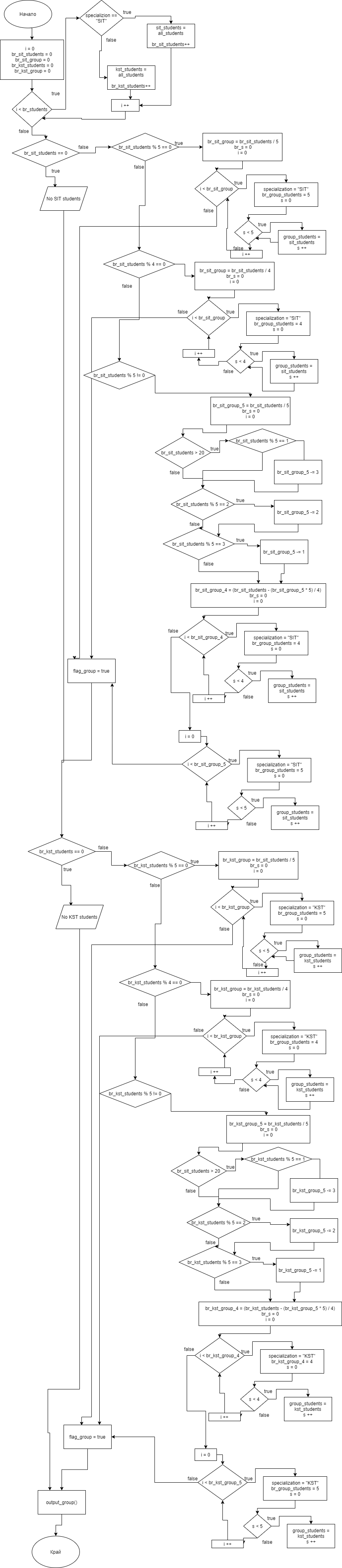
void time\_now(student\* all\_students, int& br\_students, int& year, int& month) – изчислява възраста на учениците

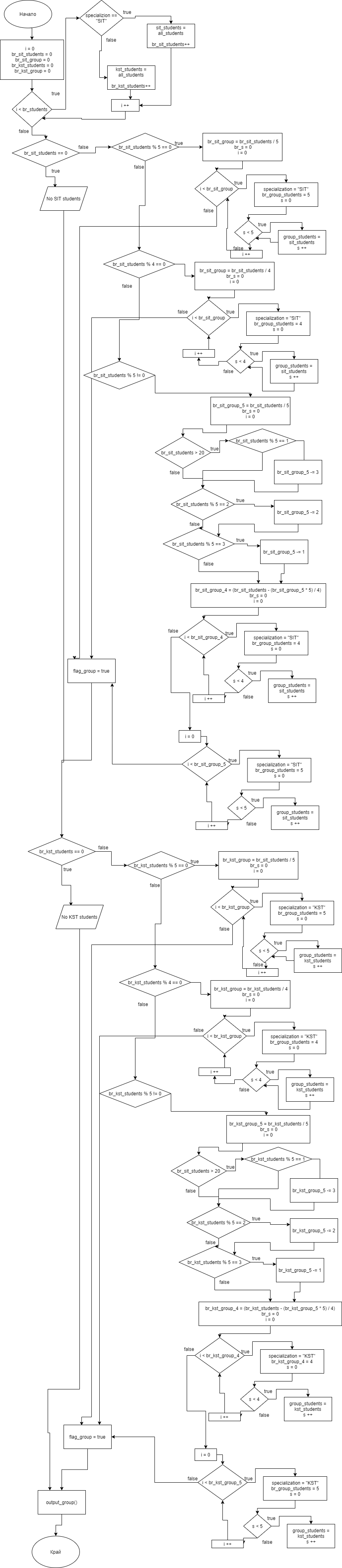


void find\_high\_GPA(student all\_students[], int br\_students) – изкарва ученици с среден успех от 5,50 и нагоре

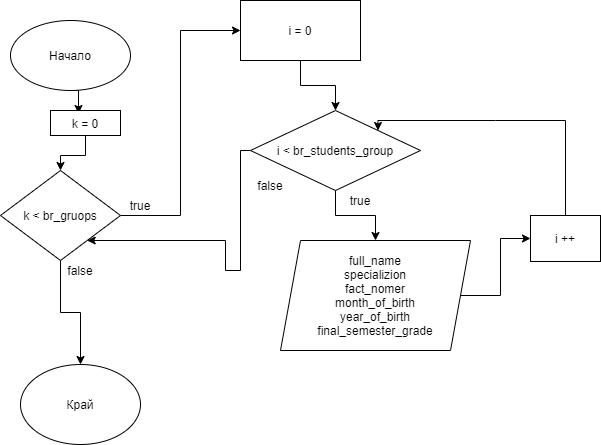


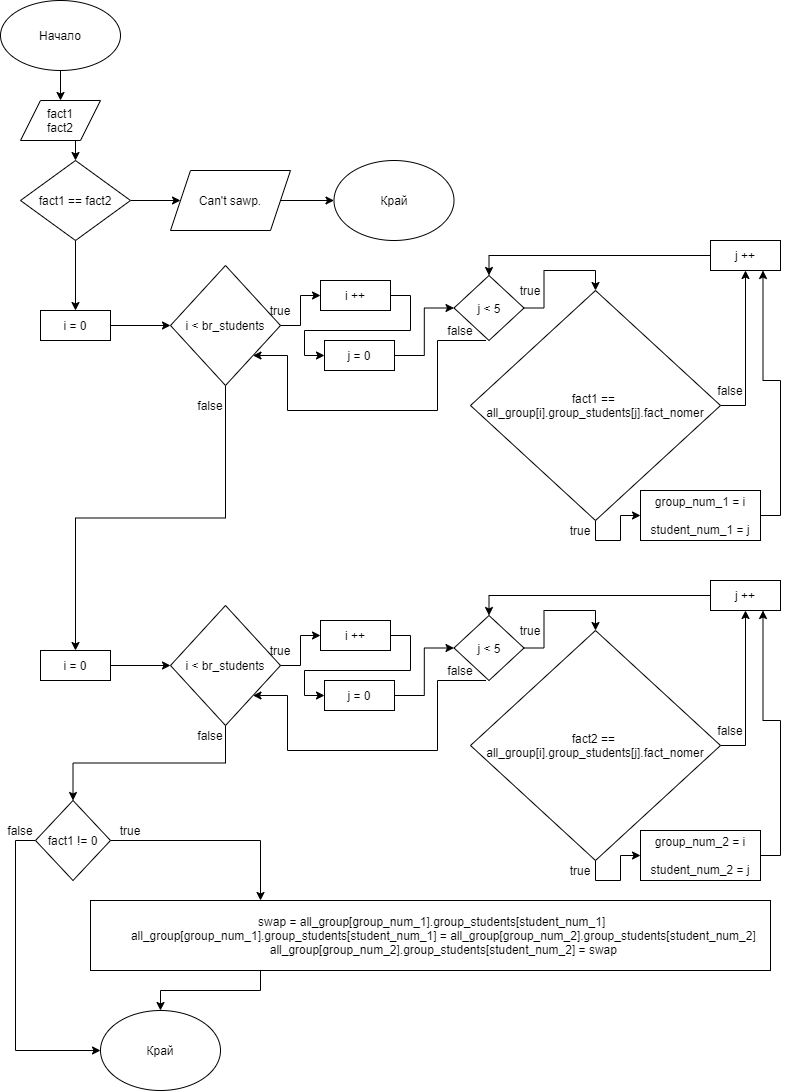
void group\_distribution(student\* all\_students, int& br\_students, group\* all\_group, int& br\_group, bool& flag\_group) – разпределя учениците по групи чрез използването на много проверки, от сорта дали брой СИТ ученици се дели на пет без остатък и ги пазпредела в друпи по пет , има и групи с по четири.





void output\_group(group all\_group[], int br\_group)- изписва на екрана групите, група по станица



void swap\_students(group\* all\_group, int& br\_group) – размена студенти по факултетни номера, проверява къде са намират учениците и ги разменя /факултетни номера на учениците/. 

void greatest\_GPA(student all\_students[], int br\_students) – извашда най-високи успех от СИТ и КСТ, намира първия от СИТ и КСТ, защото в началото се сортират по успех.

