Міністерство освіти і науки України
Національний технічний університет
«Харківський політехнічний інститут»
Кафедра «Обчислювальна техніка та програмування»

ЗВІТ
Про виконання розрахункового завдання
«Розробка інформаційно-довідкової системи»

Керівник викладач:
Бульба С.С.

Виконавець:
студент гр. КІТ-120В
Яйло Данило

Харків 2020

1.Вимоги

1.1 Розробник

Яйло Данило Дмитрович

студент групи КІТ-120В

29.04.2021

1.2 Загальне завдання

Закріпити отримані знання з дисципліни “Програмування” шляхом виконання типового комплексного завдання.

1.3 Індивідуальне завдання

У відділі орнітології почався перепис усіх зареєстрованих птахів. Вчені збирають інформацію щодо птахів. Розробити методи для роботи з колекцією:

- Знайти відсоткове відношення самок до самців у відділі.

- Знайти середній вік усіх не окольцованих птахів.

- Знайти птаха з найдовшою зимівлею.

1.4 Призначення та галузь застосування

Програма призначена для обробки вхідних даних, створення колекції птахів обробки та роботи з нею. Застосування програми призначено для людей, які працюють з базою даних птахів та їм необхідно цю базу даних впорядковувати, оновлювати та змінювати.

2. Виконання роботи

2.1 Опис вхідних та вихідних даних

Вхідні дані — файл з таблицею впорядкованих вхідних даних стосовно птахів, дані введені користувачем з клавіатури.

Вихідні дані — файл з обробленою користувачем колекцією птахів, текстові дані виведені у консоль.

2.2 Опис складу технічних та програмних засобів

Комп’ютер будь-якої потужності, IDE для написання коду програми, компілятор для обробки коду, монітор для виведення результатів роботи.

2.3 Опис джерел інформації

Офіційна документації стосовно мови розробки C++

https://docs.microsoft.com/en-us/cpp/?view=msvc-160

Інформація щодо стандартної бібліотеки шаблонів

https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D0%BD%D0%B4%D0%B0%D1%80%D1%82%D0%BD%D0%B0%D1%8F_%D0%B1%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_%D1%88%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%BE%D0%B2

2.4 Фрагменти коду функціональної частини програми програми

Див. Додаток А

2.5 UML-діаграми класів та їх зв’язків

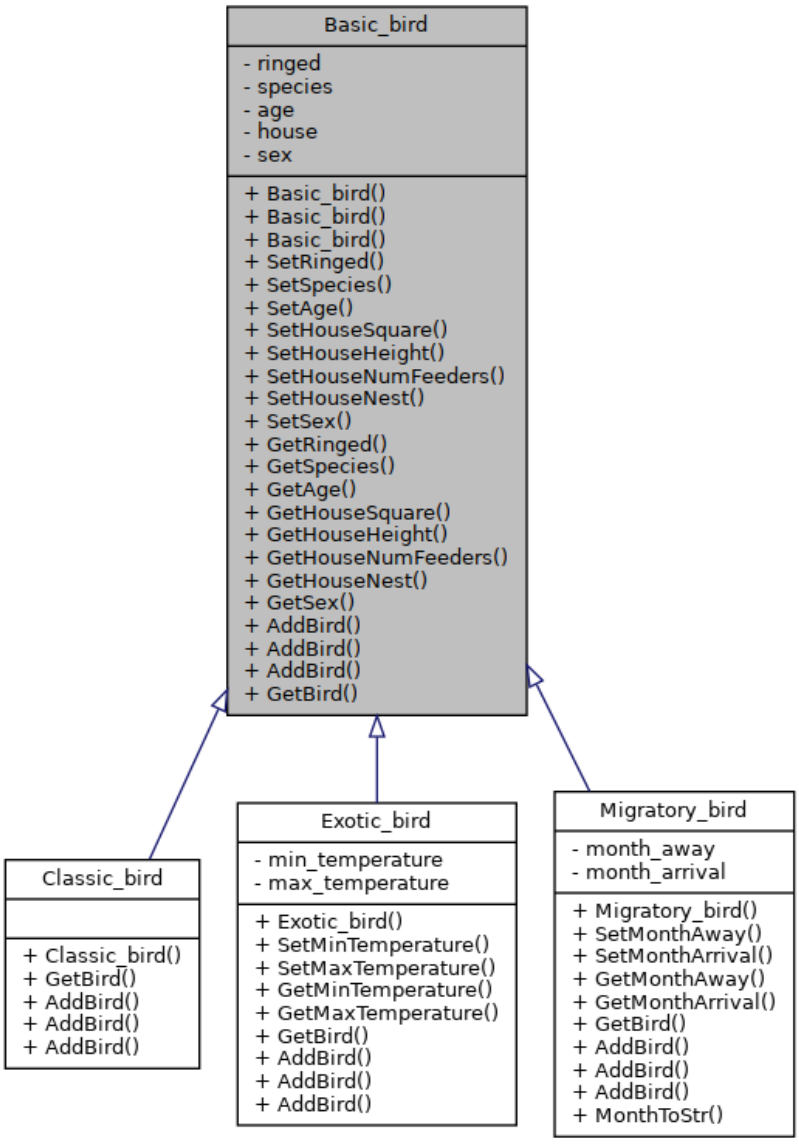

Рисунок 1.1 — *uml-діаграмма успадкувань*

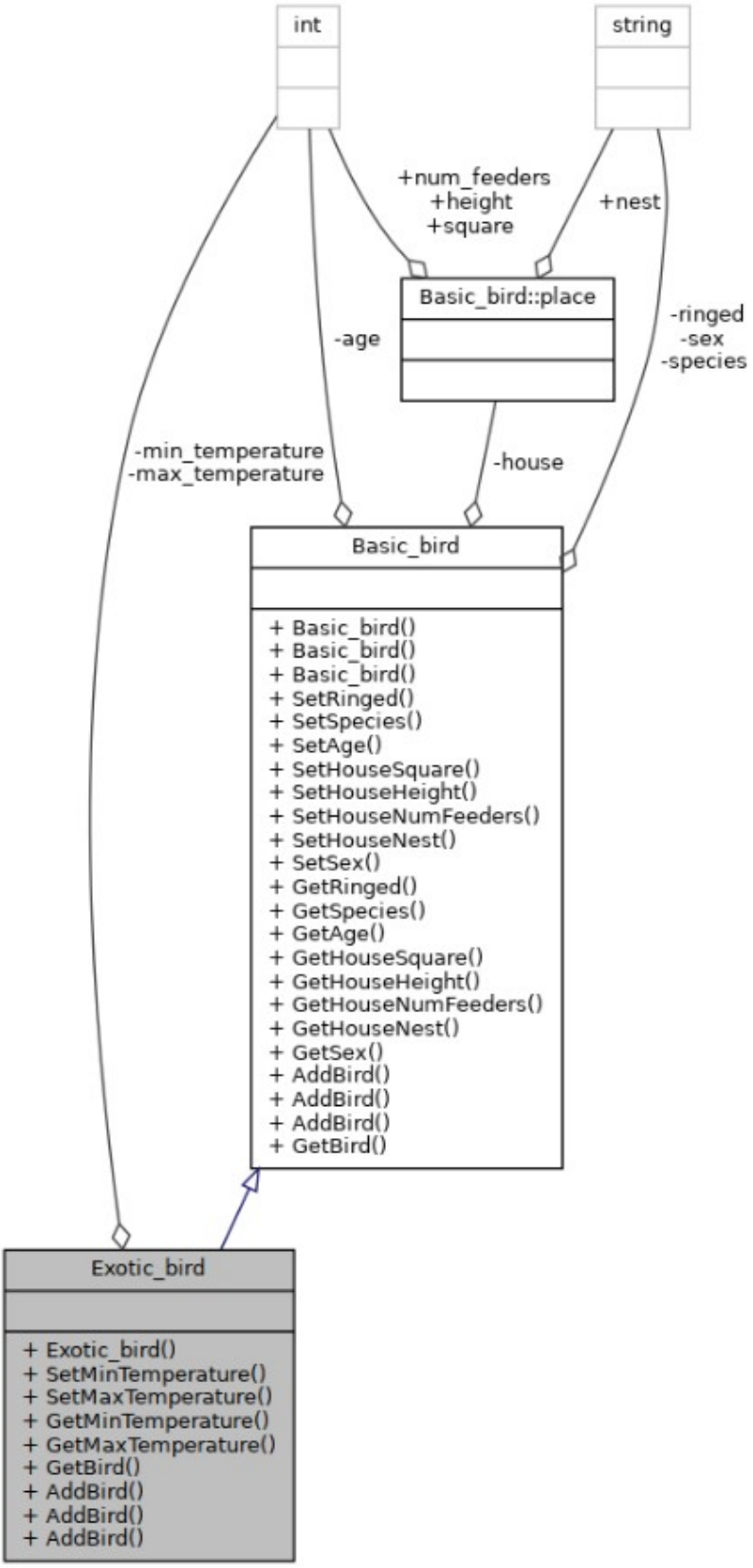

Рисунок 1.2 — uml-діаграмма зв’язків класу Exotic_bird

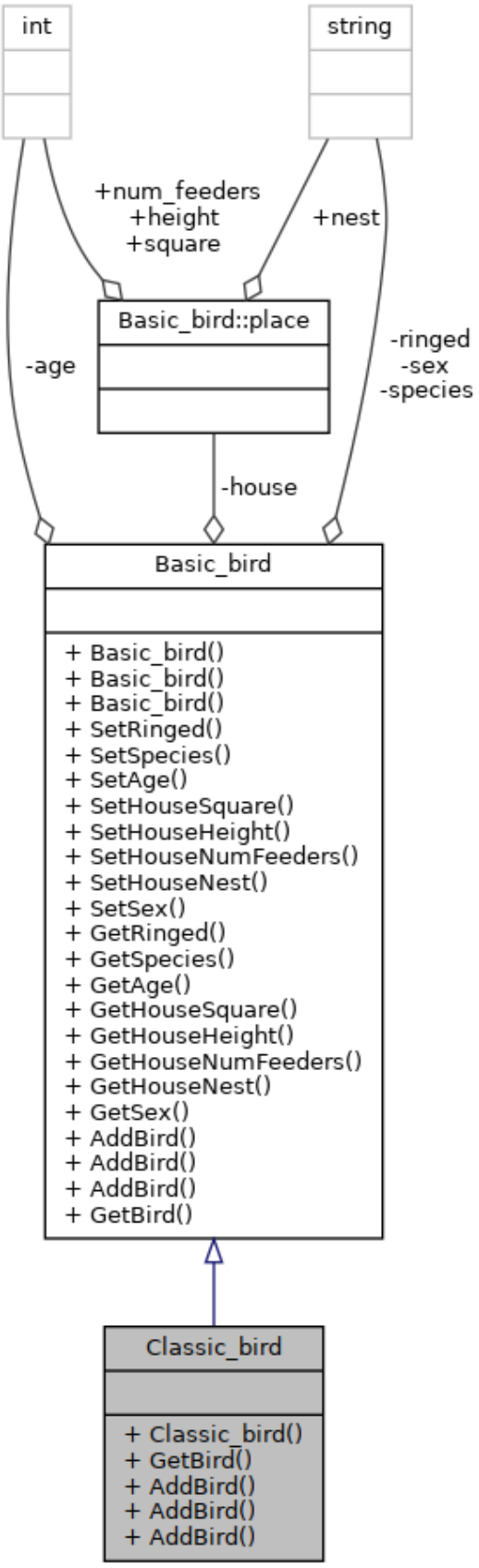

Рисунок 1.3 — uml-діаграмма зв’язків класу Classic_bird

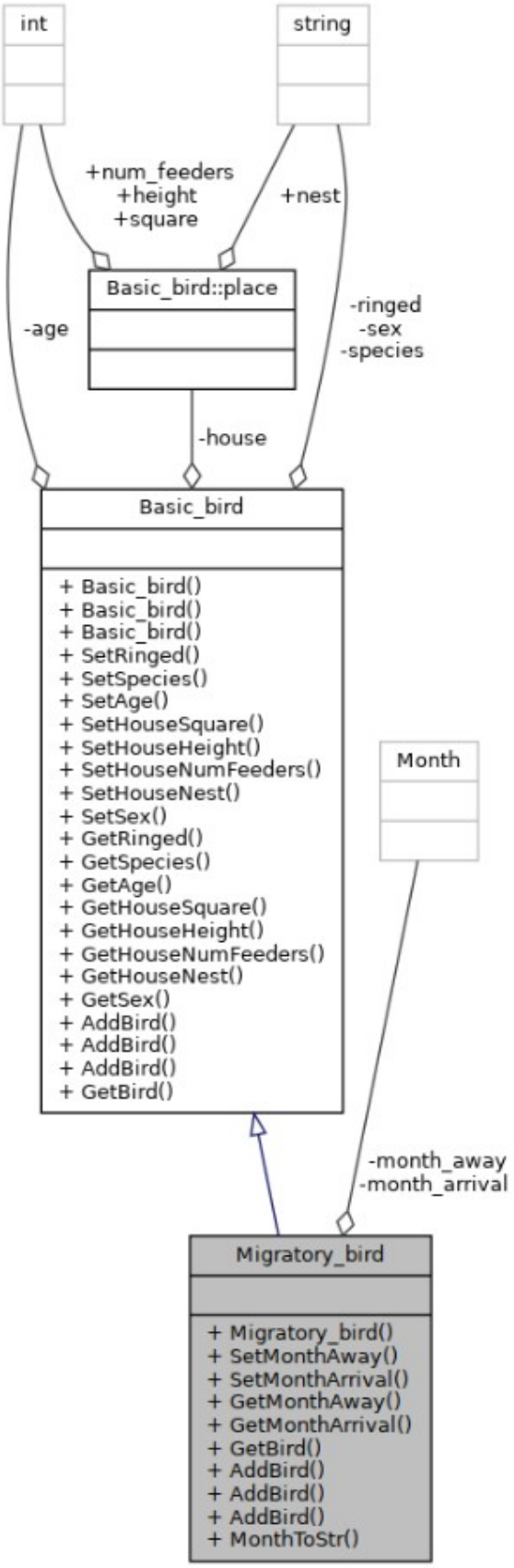

Рисунок 1.4 — uml-діаграмма зв’язків класу Migratory_bird

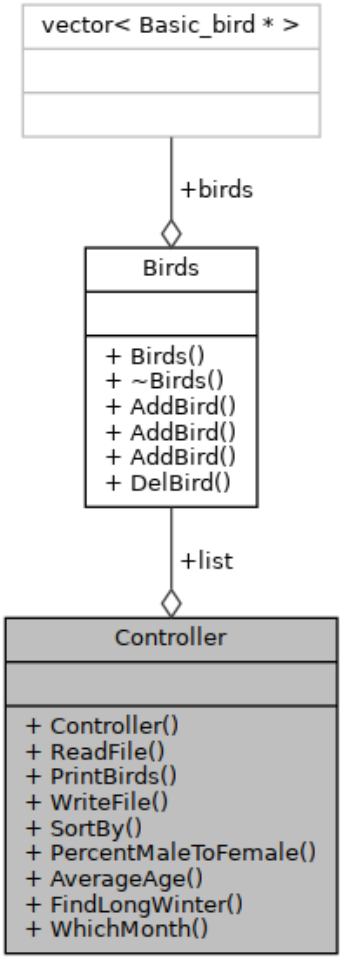

Рисунок 1.5 — uml-діаграмма зв’язків класу Birds та Controller

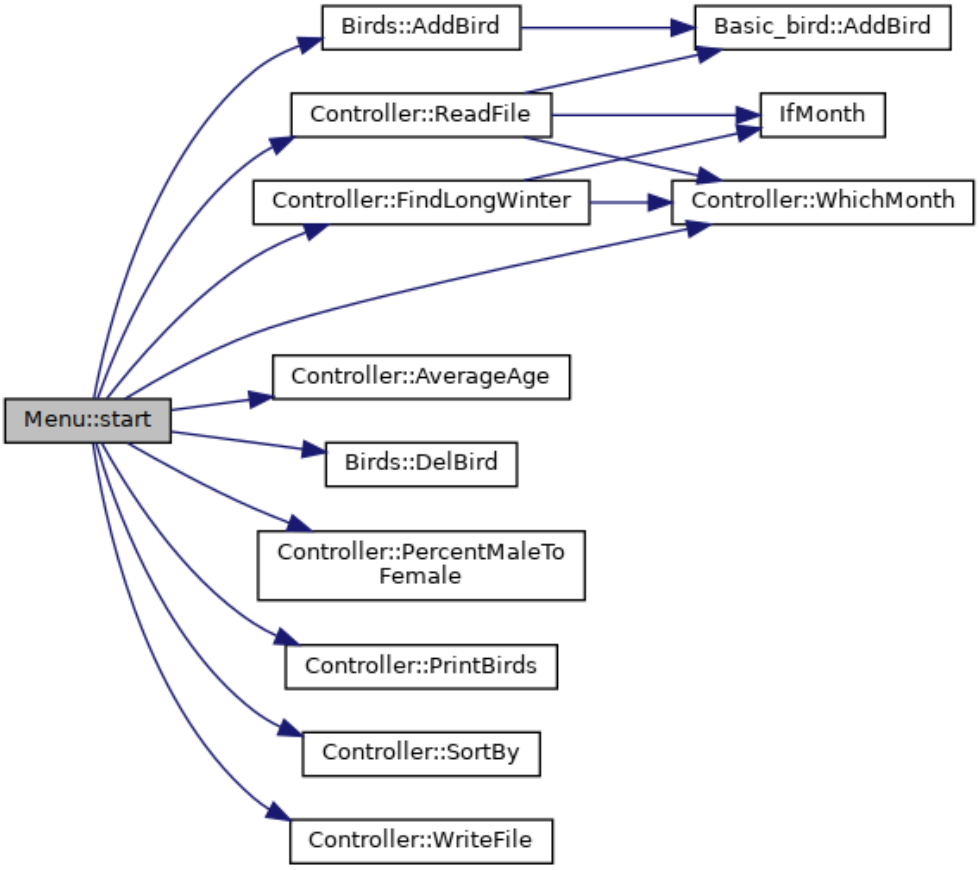

Рисунок 1.6 — граф виклику функцій класу Menu

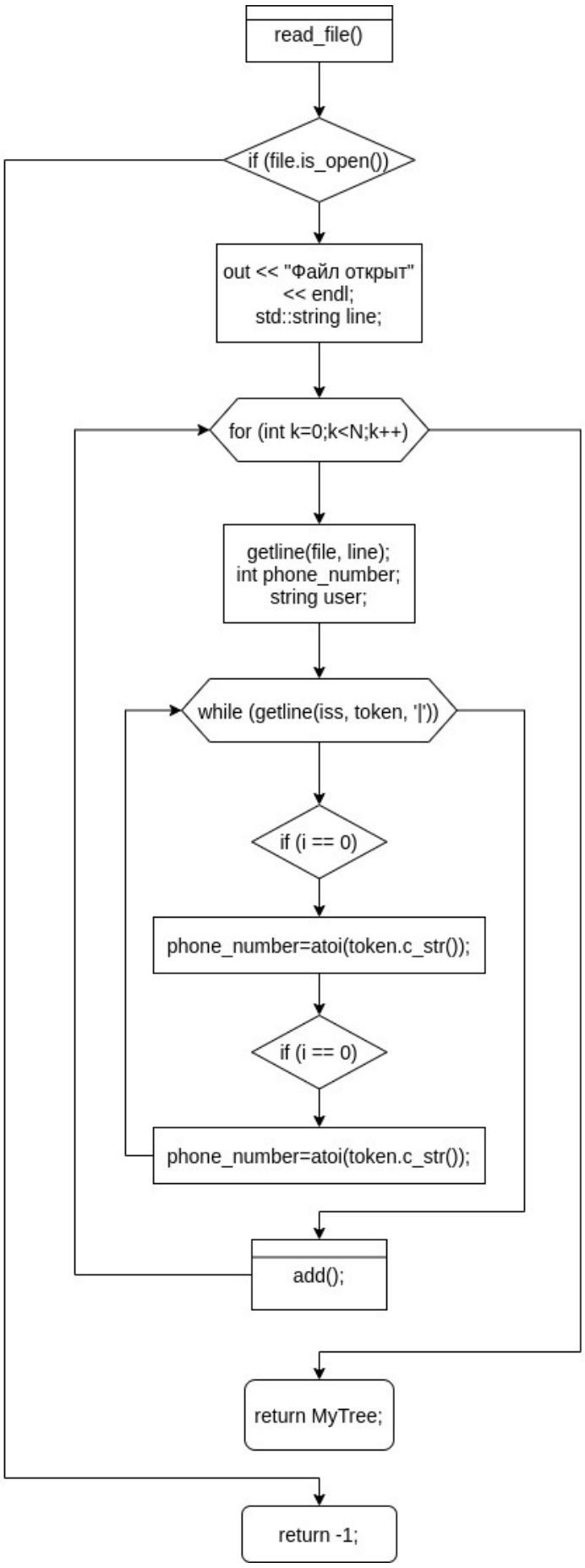

Рисунок 1.7 — блок-схема функції ReadFile

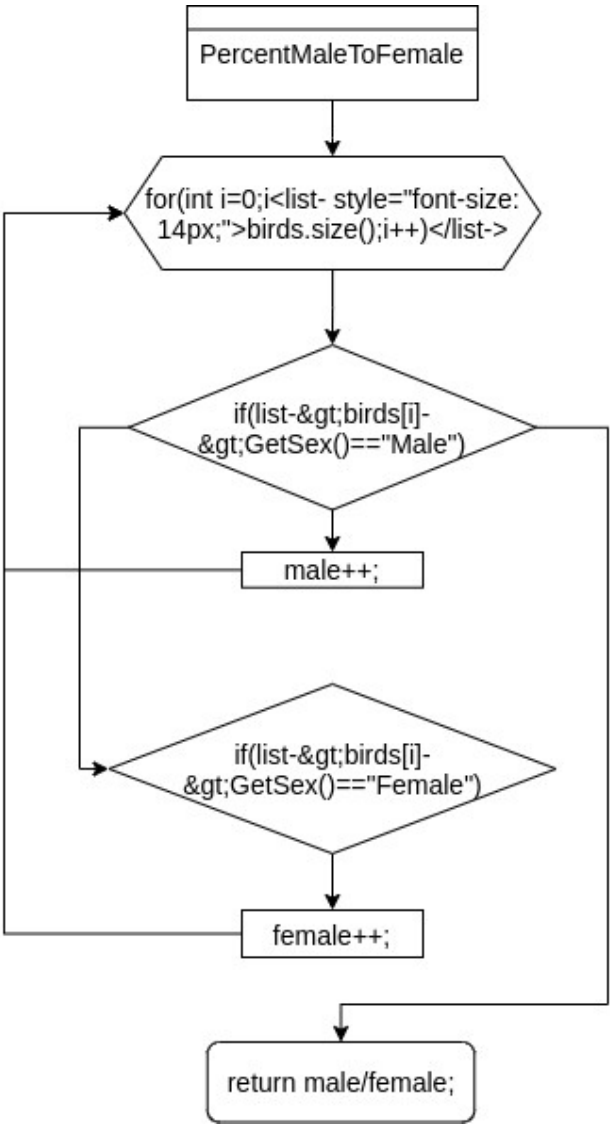

Рисунок 1.8 — блок-схема функції PercentMaleToFemale

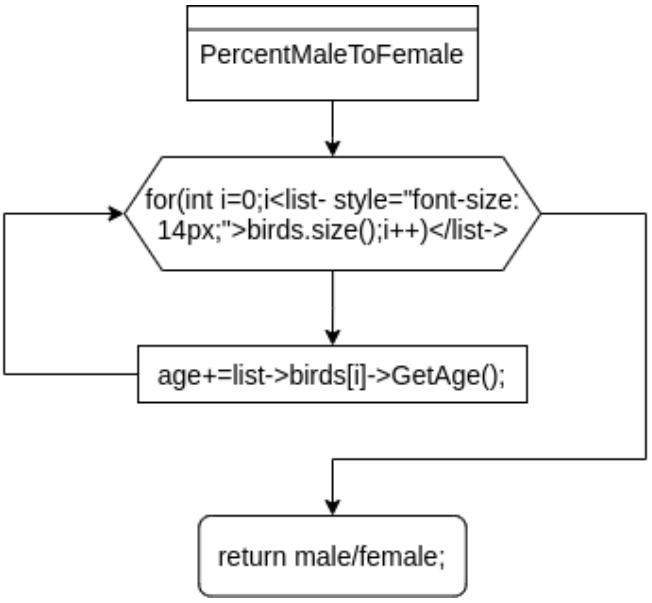

Рисунок 1.9 — блок-схема функції AverageAge

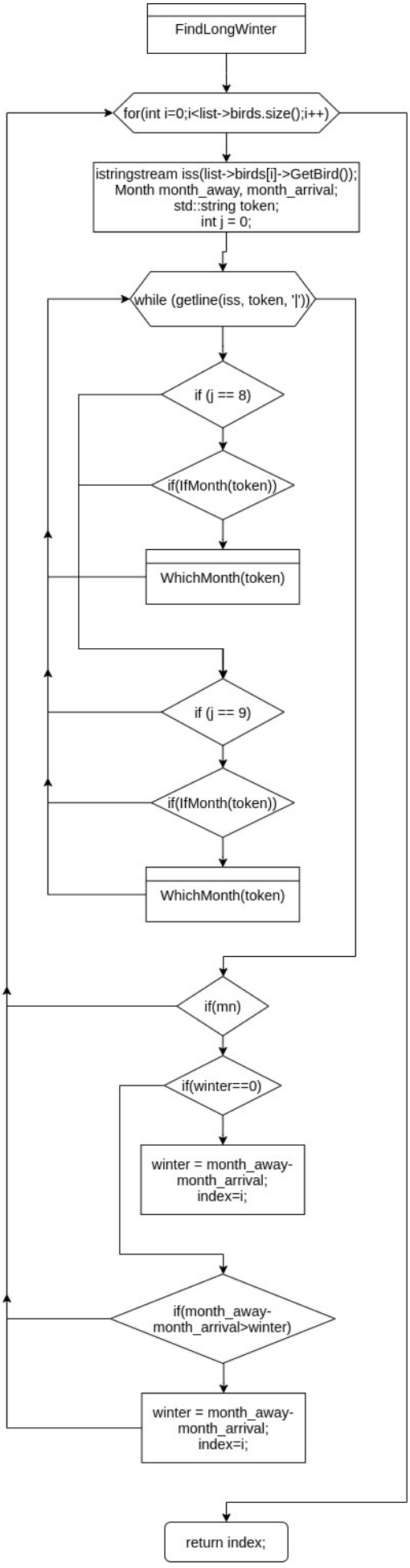

Рисунок 1.10 — блок-схема функції FindLongWinter

2.6 Перевірка програми за допомогою утіліти Valgrind

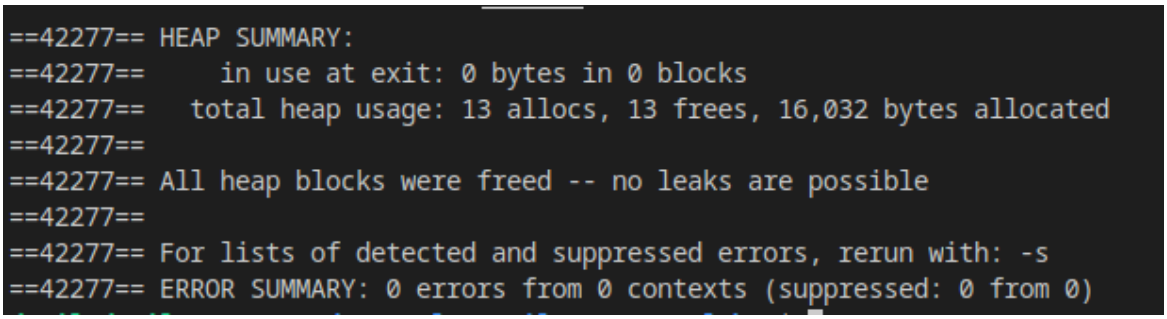

Рисунок 1.11 — утіліта Valgrind

3. Висновки

Я закріпив отримані знання з дисципліни “Програмування” шляхом виконання типового комплексного завдання. Моє індивідуальне завдання було таким: У відділі орнітології почався перепис усіх зареєстрованих птахів. Вчені збирають інформацію щодо птахів. Розробити методи для роботи з колекцією:

- Знайти відсоткове відношення самок до самців у відділі.
- Знайти середній вік усіх не окольцованих птахів.
- Знайти птаха з найдовшою зимівлею.

Я виконав завдання та розробив запропоновані методи по роботі з колекцією. Ця програма призначена для обробки вхідних даних, створення колекції птахів обробки та роботи з нею. Продемонстрував роботу програми та коректність її виконання.

```
void Controller::ReadFile(Basic_bird *exotic_bird, Basic_bird *migratory_bird, Basic_bird *classic_bird, string filename)
{
ifstream file; //поток для считывания из файла

file.exceptions(ifstream::badbit | ifstream::failbit); //Возможность самому обрабатывать ошибки потока
try //Попытка открыть файл
{
cout << "Попытка открыть файл..." << endl;
file.open(filename);
cout << "Файл успешно открыт!" << endl;
}
catch (const exception &ex)
{
cout << ex.what() << endl; //Вывод ошибки
cout << "Введите корректное имя файла: ";
cin >> filename;
cout << endl;
Controller::ReadFile(exotic_bird, migratory_bird, classic_bird, filename); //Будет рекурсировать пока пользователь не введет
корректное название файла
}

file.exceptions(ifstream::goodbit); //Из-за отсутствия этой строки компилятор ругался раньше
std::string line; //Строчка текста

static const regex text ("^[A-Z](\\w+\\s?)+");//Регулярное выражение для текста
static const regex digits ("\\d+"); //Регулярное выражение для чисел

//Будем считывать информацию построчно до тех пор,
//пока не закончится файл
while (getline(file, line))
{
//Буферные переменные в которые будет разбиваться строчка
Month month_away, month_arrival;
string ringed, species, nest, sex;
int age, square, height, num_feeders, min_temperature=0, max_temperature=0;

//Создадим поток для считывания данных из строчки
istringstream iss(line);
std::string token;
int i = 0;
while (getline(iss, token, '|'))//Разбиваем строку на блоки разделённые |
{
//В зависимости от номера блока заносим данные в соответсвующую переменную

if (i == 0){
if(regex_match(token, text)) //Если соответсвует регулярному выражению
ringed=token;
}
if (i == 1){
if(regex_match(token, text))
species=token;
}
if (i == 2){
if(regex_match(token, digits))
age=atoi(token.c_str());
}
if (i == 3){
if(regex_match(token, digits))
square=atoi(token.c_str());}
if (i == 4){
if(regex_match(token, digits))
height=atoi(token.c_str());}
if (i == 5){
if(regex_match(token, digits))
num_feeders=atoi(token.c_str());}
```

```
if (i == 6){
if(regex_match(token, text))
nest=token;}
if (i == 7){
if(regex_match(token, text))
sex=token;}
if (i == 8){
if(IfMonth(token)) //Если там текст который соответсвует типу Month
month_away=WhichMonth(token);
else
min_temperature=atoi(token.c_str()); //Иначе запись числа : запись корректных данных в таблицу на совести пользователя
}
if (i == 9){
if(IfMonth(token)) //Если там текст который соответсвует типу Month
month_arrival=WhichMonth(token);
else
max_temperature=atoi(token.c_str()); //Иначе запись числа
}
i++;
}

if(i==6) //Если было всего 6 елементов строки - базовый класс
list->birds.emplace_back(classic_bird->AddBird(ringed, species, age, square, height, num_feeders, nest, sex));
else if(max_temperature==0){ //Если числовой параметр температуры не изменился
//Пушим в вектор указатель который вернёт функция добавления элемента из обьекта класса Migratory_bird
list->birds.emplace_back(migratory_bird->AddBird(month_away, month_arrival, ringed, species, age, square, height, num_feeders,
nest, sex));
}
else{ //Если числовой параметр температуры изменился
//Пушим в вектор указатель который вернёт функция добавления элемента из обьекта класса Exotic_bird
list->birds.emplace_back(exotic_bird->AddBird(min_temperature, max_temperature, ringed, species, age, square, height,
num_feeders, nest, sex));
}
}
}



void Controller::PrintBirds(){
  cout <<endl;
  cout <<"----------------------------------------------Вывод содержимого коллекции----------------------------------------------\n"<<endl;
  for(int i=0; i<list->birds.size();i++){
  cout << "Птица №" << i << ":\n"<<list->birds[i]->GetBird()<<endl;
  }
  cout <<"-----------------------------------------------------------------------------------------------------------------------"<<endl;
  }

  int Controller::FindLongWinter(){
  int winter=0; //Для хранения наидольшей зимовки
  int index=0; //Для хранения индекса птицы с наидольшей зимовкой
  bool mn = false; //Флаг для проверки есть ли у данной птицы поля отвечающие за месяца

  for(int i=0;i<list->birds.size();i++){
  istringstream iss(list->birds[i]->GetBird());
  Month month_away, month_arrival;
  std::string token;
  int j = 0;
  while (getline(iss, token, '|'))//Разбиваем строку на блоки разделённые |
  {
  //В зависимости от номера блока заносим данные в соответсвующую переменную
  if (j == 8){
  if(IfMonth(token)) //Если там текст который соответсвует типу Month
  month_away=WhichMonth(token);
  mn=true;
  }
  if (j == 9){
```

```
if(IfMonth(token)) //Если там текст который соответсвует типу Month
month_arrival=WhichMonth(token);

}
i++;
}
if(mn){ //Если у птицы есть поля с месяцами
if(winter==0){ //Если это первая такая птица
winter = month_away-month_arrival;
index=i;
}

else if(month_away-month_arrival>winter){ //Если зимовка этой птицы больше чем предыдущей наибольшей
winter = month_away-month_arrival;
index=i;
}
mn=false;
}
}
return index; //Вывод птицы с наибольшей зимовкой
}
```