**Контрольная работа**

Контрольная работа представляет собой опрос. Список вопросов выбирается по последней цифре пароля из 140 штук по 14 вопросов каждому студенту (например, если у вас пароль заканчивается на цифру 5, то вы выбираете темы под номерами 5, 15, 25, 35, …, 135).

Контрольная работа должна содержать обязательно исходные данные (номер и текст вопроса)

**Вопросы по теме 1**

1. В чём заключается разница между компилятором и интерпретатором?
2. Какие действия производятся в фазе синтаксического анализа?
3. Какие стратегии используются компилятором для принятия решений?

**Вопросы к теме 2**

1. Что представляет собой предиктивный анализ?
2. Что означает термин «алфавит языка»?
3. Из каких частей состоят продукции грамматики?
4. Что представляют собой узлы синтаксического дерева?

**Вопросы к теме 3**

1. Для чего производится сборка мусора?
2. Какие соглашения используются для управления памятью и для чего?
3. Назовите варианты промежуточных представлений.
4. Что такое граф потока?

**Вопросы к теме 4**

1. В чём состоит преобразование "перемещение кода"?
2. Как вычисляется наибольшее количество одновременно выполняемых операций?
3. В чём заключается закон Амадаля?

**Лабораторные работы**

### Рекомендации по установке программ

**Программы**

QT (последние версии на [http://www.qt.io](http://www.qt.io/) )

**ЛАБОРАТОРНАЯ РАБОТА №1**

**«ОПРЕДЕЛЕНИЕ ВРЕМЕНИ РАБОТЫ ПРИКЛАДНЫХ ПРОГРАММ»**

**Цели работы**

1. Изучение методики измерения времени работы подпрограммы.

2. Изучение приемов повышения точности измерения времени работы подпрограммы.

3. Изучение способов измерения времени работы подпрограммы.

4. Измерение времени работы подпрограммы в прикладной программе.

**5. ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ**

1. Написать программу на языке C или C++, которая реализует выбранный алгоритм из задания.

2. Проверить правильность работы программы на нескольких тестовых наборах входных данных.

3. Выбрать размерность массива *N* такой, чтобы время работы программы более медленным алгоритмом было порядка 5 секунд.

4. Оценить (на 5-10 тестах) относительное быстродействие алгоритмов.

5. Составить отчет по лабораторной работе. Отчет должен содержать следующее:

**1.** Титульный лист.

**2.** Цель лабораторной работы.

**3.** Вариант задания.

**4.** Описание методики для определения времени работы программы.

**5.** Результат измерения времени работы программы.

**6.** Полный компилируемый листинг реализованной программы и команду для ее компиляции.

**7.** Вывод по результатам лабораторной работы.

**6. ВАРИАНТЫ ЗАДАНИЙ**

Таблица вариантов

|  |  |  |
| --- | --- | --- |
| № | Алгоритм 1 | Алгоритм 2 |
| 1 | Пузырьком | Шелла |

В качестве справочной информации по алгоритмам можно использовать, например, соответствующие статьи "Википедии" (<https://ru.wikipedia.org/wiki/>)

**ЛАБОРАТОРНАЯ РАБОТА №2.**

**«ИЗУЧЕНИЕ ОПТИМИЗИРУЮЩЕГО КОМПИЛЯТОРА»**

**Цели работы**

1. Изучение основных функций оптимизирующего компилятора, и некоторых примеров оптимизирующих преобразований и уровней оптимизации.

2. Получение базовых навыков работы с компилятором GCC.

3. Исследование влияния оптимизационных настроек компилятора GCC на время исполнения программы.

В лабораторной работе описаны основные возможности компилятора GCC/CNU, являющегося классическим образцом оптимизирующего компилятора. При выполнении задания в ОС Windows рекомендуется QT (<https://www.qt.io/>) - бесплатный IDE для C++ с компилятором MinGW (программный порт GCC под Windows).

**ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ**

1. Написать программу на языке C или C++, которая реализует выбранный алгоритм из задания.

2. Проверить правильность работы программы на нескольких тестовых наборах входных данных.

3. Выбрать значение параметра *N* таким, чтобы время работы программы было порядка 30-60 секунд.

4. Программу скомпилировать компилятором GCC с уровнями оптимизации -**O0**, -**O1**, -**O2**, -**O3, -Os, -Ofast, -Og** под архитектуру процессора x86.

5. Для каждого из семи вариантов компиляции измерить время работы программы при нескольких значениях *N***.**

6. Составить отчет по лабораторной работе. Отчет должен содержать следующее.

**1.** Титульный лист.

**2.** Цель лабораторной работы.

**3.** Вариант задания.

**4.** Графики зависимости времени выполнения программы с уровнями оптимизации -**O0**, -**O1**, -**O2**, -**O3, -Os, -Ofast, -Og** от параметра *N*.

**5.** Полный компилируемый листинг реализованной программы и команды для ее компиляции.

**6.** Вывод по результатам лабораторной работы.

**7. ВАРИАНТЫ ЗАДАНИЙ**

Варианты заданий взять из лабораторной работы №1.

|  |  |  |
| --- | --- | --- |
| № | Алгоритм 1 | Алгоритм 2 |
| 1 | Пузырьком | Шелла |

**Лабораторная работа № 3.**

**Оптимизация временной сложности алгоритма**

**Цели работы**

1. Изучение основных понятий, связанных с оценкой временной сложности алгоритмов .

2. Изучение практических приемов оптимизации кода.

**ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ**

1. В соответствии с вариантом задания, реализовать алгоритмы так, чтобы их временная сложность соответствовала требуемой в постановке задачи или были проведены иные оптимизации (см. табл. 2). Решения представить в виде работающих программ на C++.

Таблица 2. Варианты задания

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Алгоритм | Реализация на C++ | Текущая временная сложность кода | Требуемая временная сложность кода и/или улучшение алгоритма |
| 1 | Выбор элементов главной диагонали квадратной матрицы A[n,n] | for (int i=0; i<n; i++)  for (int j=0;j<n; j++)  if (i==j)  cout << a[i][j] << " "; | *O*(*n*2) | *O*(*n*) |
| 10 | Поиск в массиве натуральных чисел a[n] пары элементов со значениями item1 и item2, расстояние между которыми минимально | const int n=10;  int a[n] = {  3,5,10,8,4,12,5,3,0,7  };  int item1=3,item2=8,min=n;  for (int i=0; i<n; i++)  for (int j=0; j<n; j++)  {  if (a[i]==item1 &&  a[j]==item2 &&  abs(i-j)<min) {  min = abs(i-j);  }  }  cout << min; | *O*(*n*2) | *O*(*n*) |

2. Составить отчет по лабораторной работе. Отчет должен содержать следующее.

**1.** Титульный лист.

**2.** Цель лабораторной работы.

**3.** Вариант задания.

**4.** Полный компилируемый листинг реализованной программы.

**5.** Вывод по результатам лабораторной работы.

**3. ВАРИАНТЫ ЗАДАНИЙ**

Варианты заданий взять из табл. 2.

Таблица 2. Варианты заданий

|  |  |  |
| --- | --- | --- |
| № вар. | Алгоритм 1 | Алгоритм 2 |
| 1 | 1 | 10 |