Балтийский государственный технический университет «ВОЕНМЕХ» им. Д. Ф. Устинова

Кафедра И5  
«Информационные системы и программная инженерия»

**Лабораторная работа № 1** по дисциплине «Компьютерный практикум»на тему **«Особенности использования набора компиляторов и утилит GCC»**

**Выполнил:**

Студент Селюхов М.М  
Группа И582

**Преподаватель**:  
 Вальштейн К. В.

Санкт-Петербург  
2019

Задача 1.

Дано натуральное число N. Составить программу для сравнения цифр старшего и младшего разрядов этого числа. Провести поэтапную компиляцию исходного текста написанной программы, разобраться в результатах, полученных на каждом этапе компиляции. Провести оптимизацию кода написанной программы с помощью набора компиляторов GCC, пояснить внесённые для оптимизации кода изменения.

Создадим файл исходного текста программы на языке C++ с использованием текстового редактора «Dev-Cpp» и сохраним его под именем «1.cpp» в рабочем каталоге «C:\ccomp\» .

*Текст программы:*

#include <cstdlib>

#include <cstdio>

#include <iostream>

using namespace std;

#include <stdio.h>

int main()

{

int first, last, N;  
   
 printf("Enter N numeric\n");  
 scanf("%d", &N);

if ((N/10)==0)  
 {  
 printf("Error.");  
 return 0;  
 }  
   
 first = N;  
 do  
 {  
 first=first/10;

}  
 while(first>9);  
   
 last = N%10;  
  
 if(first>last) printf("Max first");  
 else if(last>first) printf("Max last");  
 else printf("last = first");  
 return 0;  
}  
*Результаты тестирования:*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый  результат** | **Результат работы программы** |
| 12456267 | "Max last" | "Max last" |
| 896 | "Max firs" | "Max first" |

Применив команду **gcc -E -o num.i num.c** мы создадим файл num.i, содержащий исходный текст, обработанный препроцессором. В данный файл будет добавлено содержимое заголовочных файлов, будут удалены комментарии и раскрыты макросы.

Применив команду **-S num.c -o num.s** . мы получим файл с ассемблерным кодом, соответствующим исходному тексту.

Результатом команды **-c num.c -o num.o** будет объектный файл, содержащий блоки готового к исполнению машинного кода, блоки данных, а также список определенных в файле функций и внешних переменных, но при этом в нем не заданы абсолютные адреса ссылок на функции и данные. Объектный файл не может быть запущен на исполнение непосредственно, но в дальнейшем может быть объединен с другими объектными файлами.

G++ позволяет провести оптимизацию кода во время компиляции. Применим команды:

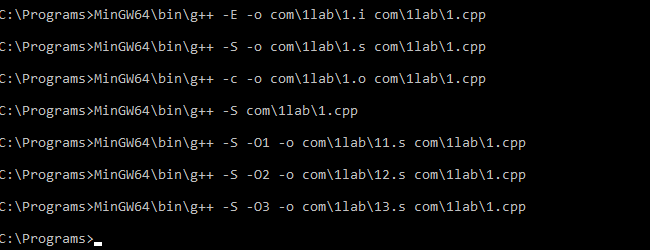
**gcc -S -O1 -o num1.s num.c**

**gcc -S -O2 -o num2.s num.c**

**gcc -S -O3 -o num3.s num.c**

После этого в файлах «num1.s», «num2.s» и «num3.s» будет код с различной степенью оптимизации.

Проанализируем размеры полученных файлов «num1.s» - 1,38 Кб, «num2.s» - 1,55 Кб и «num3.s» -1,55 Кб. Таким образом, видно, что оптимизация с уровнем –О1 позволяет сгенерировать наименьший по объему код, но оптимизация O2 приводит к менее массивному коду с наименьшим числом операций и переходов по меткам.



Задача 2.

При помощи любого текстового редактора написать две программы согласно следующим условиям:

1. Определить количество элементов массива М (22), больших среднего арифметического значения элементов этого массива.

2. В массиве А (45) найти локальные максимумы, определить их местоположение. Локальным максимумом назовем элемент массива, значение которого больше, чем значения двух соседних с ним элементов.

Функции для работы с массивом вынести в отдельные файлы: в одном файле описать функции для ввода/вывода массива, в другом - для обработки массива. В обеих программах должны использоваться одни и те же функции для ввода/вывода массивов, описанные в одном из этих файлах. Провести раздельную компиляцию написанных файлов. Скомпилировать обе программы, используя созданные объектные файлы, обе программы должны использовать один и тот же объектный файл с функциями для ввода/вывода массива. Создать статическую библиотеку для ввода/вывода массива и продемонстрировать возможности по её подключению.

Создадим файл исходного текста первой программы на языке C с использованием текстового редактора «Notepad++» и сохраним его под именем «main.c» в рабочем каталоге «C:\compp\» .

*Текст первой программы:*

*main.c*

#include <stdio.h>  
 #include "input\_print.h"  
 #include "count.h"  
   
 #define SIZE 22  
   
 int main(void)  
 {  
 int arr[SIZE];  
 array\_input(arr, SIZE);  
 count(arr, SIZE);  
 return 0;  
 }

Создадим файл исходного текста второй программы на языке C с использованием текстового редактора «Dev-Cpp» и сохраним его под именем «main.cpp» в рабочем каталоге «C:\compp\» .

*Текст второй программы:*

*main2.c*

#include <stdio.h>  
 #include "input\_print.h"  
 #include "count.h"  
   
 #define SIZE 22  
   
 int main(void)  
 {  
 int arr[SIZE];  
 array\_input(arr, SIZE);  
 locmin(arr, SIZE);  
 return 0;  
 }

Создадим отдельный файл согласно условиям задания, содержащий функции для обработки массива, на языке C с использованием текстового редактора «Notepad++» и сохраним его под именем «input\_print.c», «count.c», «locmin.c» в рабочем каталоге «C:\compp» .

*Текст программы first.cpp:*

include "stdio.h"  
#include "input\_print.h"

void array\_print(int\* arr, unsigned int c)  
{  
 int i;  
 printf("Array:\n");  
 for (i = 0; i < c; i++)  
 {  
 printf("%d ", arr[i]);  
 }  
 printf("\n");  
}

void array\_input(int\* arr, unsigned int c)

{  
 int i;  
 srand(time(NULL));  
 printf("Fill Array: ");  
 for (i = 0; i < c; i++)  
 {  
 scanf("%d", &arr[i]);  
 }  
}

*Текст программы count.c:*

#include "stdio.h"   
 #include "count.h"

void count(int\* arr, unsigned int c)   
 {  
 int i,sum=0, k = 0;  
 float S;  
 for(i=0; i<c; i++) sum+=arr[i];  
 S = sum/c;  
 for(i=0; i<c; i++) if(arr[i]>S) k+=1;   
 printf("Count = %d", k);  
 }  
*Текст программы locmin.c:*

#include "stdio.h"  
 #include "locmin.h"

void locmin(int\* arr, unsigned int c)  
 {  
 int i;  
 printf("local min nubmer:\n");  
 for(i=1; i<c-1; i++)  
 if ((arr[i]<arr[i-1])&&(arr[i]<arr[i+1]))  
 printf("%d, ", i+1);  
 }

Используя команду

Была проведена раздельная компиляция каждого из файлов, с помощью команд описанных в 1 части. В итоге были получены 5 объектных файлов: main.o, main2.o, count.o, localmin.o, input\_print.o. Данные объектные файлы были собраны в исполняемые файлы для выполнения задач 1 и 2.

*Результаты тестирования первой программы при раздельной компиляции:*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый  результат** | **Результат работы программы** |
| 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 | 11 | 11 |

*Результаты тестирования второй программы при раздельной компиляции:*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый  результат** | **Результат работы программы** |
| 2 1 2 3 1 2 4 55 6 77 4 6 12 55 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 3 4 5 4 3 2 4 5 6 7 7 8 6 5 4 | 2, 5, 9, 11, 15, 17, 19, 21, 23, 25, 27, 29, 36, | 2, 5, 9, 11, 15, 17, 19, 21, 23, 25, 27, 29, 36, |

Перейдем к созданию статической библиотеки для ввода/вывода массива. Создадим в рабочем каталоге «C:\Programs\com\1.2\» заголовочный файл «lib.h», содержащий прототипы функций из файла «count.o», «input\_print.o» «locmin.o»

Использую команду

ar crs liblib.a count.o input\_print.o locmin.o

создадим библиотеку, содержащую объектный файл «count.o», «input\_print.o» «locmin.o». Для программы **ar** заданы следующие опции:  
1. Опция **c arname** - создать архив, если архив с именем **arname** не существует, он будет создан, в противном случае файлы будут добавлены к имеющемуся архиву.  
2. Опция **r** - задает режим обновления архива, если в архиве файл с указанным именем уже существует, он будет удален, а новый файл дописан в конец архива.  
3. Опция **s** - добавляет (обновляет) индекс архива. Индекс архива необходим для ускорения работы с библиотекой - для того чтобы найти нужное определение, отпадает необходимость просматривать таблицы символов всех файлов архива, можно сразу перейти к файлу, содержащему искомое имя.

Продемонстрируем возможность подключения библиотеки на примере компиляция первой программы. Добавим в текст файла «main.cpp» и «main.cpp» подключение библиотеки с помощью директивы «#include "lib.h"». Создадим объектный файл «main.o» и «main.o» c помощью команды

gcc -c -o main.c

gcc -c -o main2.c

Для компиляции с использованием библиотеки используем следующую команду:

gcc -o main.exe main.o –L llib,

опция -lname позволяет подключить библиотеку с именем libname, в описанном случае в качестве name указано hworld, такое же имя имеет подключаемый заголовочный файл. Опция -L <путь> передаёт путь к библиотеке.

*Результаты тестирования первой программы при подключении статической библиотеки:*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый  результат** | **Результат работы программы** |
| 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 | 11 | 11 |

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый  результат** | **Результат работы программы** |
| 2 1 2 3 1 2 4 55 6 77 4 6 12 55 1 2 1 2 1 2 1 2 1 2 1 2 1 2 1 2 3 4 5 4 3 2 4 5 6 7 7 8 6 5 4 | 2, 5, 9, 11, 15, 17, 19, 21, 23, 25, 27, 29, 36, | 2, 5, 9, 11, 15, 17, 19, 21, 23, 25, 27, 29, 36, |

