

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

"Московский технологический университет"

МГТУ МИРЭА

Институт информационных технологий

Кафедра ИТС

Лабораторная работа №3

по дисциплине «Программная организация

высокопроизводительных вычислений»

Преподаватель:

Торхов Алексей Евгеньевич

Исполнитель:

Пыжов Владислав Олегович

Группа: ИКМО-2-15

Москва 2016

# Постановка задачи

Цель данной лабораторной работы состоит в изучении процесса создания динамических библиотек и их применения в различных приложениях. Она реализуется посредством решения целого ряда задач.

1. Динамическая библиотека должна решать задачу по нахождению минимального значения функционала , где – целочисленный массив, содержащий значения в диапазоне , массив той же длины, содержащий элементы и 1.

При этом на используемый алгоритм накладываются ограничение по памяти и по времени выполнения.

2. Динамическая библиотека должна подключаться к Python файлу, который позволяет через параметры командной строки указать способ его ввода, а именно либо введя в ней последовательность чисел – исходного массив , либо указав файл, содержащий их.

Вывод ошибок должен осуществляться только при указании параметра «--debug», причём ошибки должны отправлять в системный журнал syslog.

3. Динамическая библиотека должна быть прилинкована к исполняемому файлу, написанному на C или C++ и демонстрирующему её работу.

# 1. Разработка динамической библиотеки

## Краткое описание алгоритма и его реализация

Для решения исходной задачи она интерпретируется в терминах задачи о разбиение множества на два подмножества с одинаковой суммой (partition problem).

Действительно, исходный массив можно преобразовать в массив положительных чисел расставив соответствующим образом знаки в нём. Затем, разбив исходное множество на два подмножества с минимальной по модулю разностью сумм, достаточно вновь изменить знаки у одной из половин на противоположные и, таким образом, получить массив , дающий искомый минимум функционала .

Однако поскольку задача не состоит в нахождении массива, достаточно найти ту упомянутую ранее минимальную разность сумм. Это и выполняет следующий алгоритм, реализованный на языке C++:

**lib.cpp**

|  |
| --- |
| #include <stdlib.h>  #include <limits.h>  extern "C" int calculateMinimum(int len, int\* array) {  // Calculates the sum of the absolute values of array elements  int sum = 0;  for (int i = 0; i < len; i++) {  sum += abs(array[i]);  }  // Builds the array of sums, which realizability we will test:  int \*available = new int[sum+1];  for (int i = 0; i < sum+1; i++) {  available[i] = false;  }  available[0] = true;  // Checks the realizability of subsets sums:  for (int i = 0; i < len; i++) {  for (int j = sum - abs(array[i]); j >= 0; j--) {  if (available[j]) {  available[j + abs(array[i])] = true;  }  }  }  int minimum = INT\_MAX;  // For all sums:  for (int i = 0; i < sum; i++) {  // If sum is realizabile  if (available[i]) {  if (abs(sum - 2\*i) < minimum) {  minimum = abs(sum - 2\*i);  }  }  }  return minimum;  } |

Для компиляции и компоновки написанной библиотеки используется следующая команда:

|  |
| --- |
| g++ -shared –o lib.so lib.cpp |

# 2. Применение динамической библиотеки

## Программа на Python

Описываемая программа служит интерфейсом для разработанной библиотеки, предоставляя интерфейс для ввода исходного массива и отображения результатов выполнения соответствующей функции этой библиотеки.

|  |
| --- |
| import ctypes, os, argparse, syslog  from ctypes import \*  def calculate(numbers):  # Converts the list of numbers to ctype array:  arr = (ctypes.c\_int \* len(numbers))(\*numbers)  # Gets the library:  dll = CDLL('lib.so')  # Call the library:  return dll.calculateMinimum(c\_int(len(arr)),byref(arr))  if \_\_name\_\_ == '\_\_main\_\_':  # The prefernces for the argument parser  parser = argparse.ArgumentParser(description=  'The simple program, that calculates the sum of integers that has the minimum  absolute value')  group = parser.add\_mutually\_exclusive\_group(required=True)  group.add\_argument('--numbers', metavar='n', type=int, nargs='+', help='set of integers')  group.add\_argument('--file', metavar='f', help='path to file with numbers')  parser.add\_argument('--debug', dest='debug', action='store\_true',  help='enable the log to syslog')  # Gets the arguments  args = parser.parse\_args()  try:  # Load the numbers from file:  if (args.file is not None):  f = open(args.file)  numbers = list(map(int,f.read().split()))  # Get the inputted numbers  else:  numbers = args.numbers  # Calculates the sum  s = calculate(numbers)  print('Минимальная по модулю сумма элементов: {0}'.format(s))  except Exception as e:  if args.debug == True:  syslog.syslog(str(e))  print('Произошла ошибка: выполнение программы остановлено') |

Для тестирования корректности результатов был написан соответствующей тест-кейс:

|  |
| --- |
| import unittest, main, itertools, sys, random  class CalculatorTests(unittest.TestCase):  # Test a calculating the sums:  def test\_sums(self):  # Calculates the sum:  def sign\_sum(array, signs):  s = 0  for i in range(len(array)):  s += array[i] \* signs[i]  return s  # Checks the different lengths  for N in range(1, 20, 1):  for j in range(0,10):  # Checks the different arrays  array = [random.randint(-100,100) for x in range(N)]  # Generates the permutations:  minimum = sys.maxsize  multiple\_signs = itertools.product([-1,1], repeat=N)  # Checks the real minimum sum  for sign in multiple\_signs:  s = abs(sign\_sum(array, sign))  if s < minimum:  minimum = s  # Checks the answer  self.assertEqual(minimum, main.calculate(array))    if \_\_name\_\_ == '\_\_main\_\_':  unittest.main() |

## Программа на C++

Для демонстрации принципа работы динамической линковки в C++ была написана простая программа, которая позволяет ввести в командной строке последовательность чисел – элементов массива, и, применяя функцию из динамической библиотеки, вычисляет искомое значение и выводи его на экран.

Код программы представлен далее:

**main.cpp**

|  |
| --- |
| #include <iostream>  #include <stdlib.h>  extern "C" int calculateMinimum(int len, int\* array);  int main(int argc, char \*\*argv) {  int \*array = new int[argc];  //There is no checking for errors, just for sake of my lovely time:  for (int i = 1; i < argc; i++) {  array[i] = atoi(argv[i]);  }  if (argc > 1) {  std::cout << "The sum: " << calculateMinimum(argc, array) << std::endl;  }  else {  std::cout << "No numbers were inputed :(" << std::endl;  }  } |

Для компиляции и компоновки используется следующая команда:

|  |
| --- |
| g++ main.cpp -o main -l:lib.so |