Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Коршун Н.И.

ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Лабораторная работа №7**

**Задание 1.**

1. Разработайте приложение **OS08\_01**.
2. Приложение **OS08\_01** выводит на консоль текущую локальную дату и время в формате ***дд.мм.ггг чч:мин:сек***.

|  |  |
| --- | --- |
| Код программы OS07\_01 | #define \_CRT\_SECURE\_NO\_WARNINGS 1  #include <iostream>  #include <ctime>  #include <iomanip>  int main() {  // Получаем текущее время  std::time\_t t = std::time(nullptr);  std::tm time\_info;  // Используем безопасную версию localtime  localtime\_s(&time\_info, &t);  // Форматируем вывод  std::cout << std::setfill('0') << std::setw(2) << time\_info.tm\_mday << "."  << std::setfill('0') << std::setw(2) << 1 + time\_info.tm\_mon << "."  << 1900 + time\_info.tm\_year << " "  << std::setfill('0') << std::setw(2) << time\_info.tm\_hour << ":"  << std::setfill('0') << std::setw(2) << time\_info.tm\_min << ":"  << std::setfill('0') << std::setw(2) << time\_info.tm\_sec << std::endl;  return 0;  } |
| Скриншот запуска |  |

**Задание 2.**

1. Разработайте приложение **OS08\_02,** выполняющее бесконечный цикл.
2. В теле цикла подсчитывается количество итераций.
3. Выведите на консоль значения счетчика итераций через 5 сек. и 10 сек.
4. Корректно завершите работу цикла и приложения через 15 сек., выведите итоговое значение счетчика итераций.

|  |  |
| --- | --- |
| Код программы OS07\_02 | #include <iostream>  #include <chrono>  #include <thread>  int main() {  setlocale(LC\_ALL, "ru");  int counter = 0;  // Бесконечный цикл  while (true) {  counter++;  // Вывод значения счетчика итераций через 5 секунд  if (counter == 5) {  std::cout << "Значение счетчика через 5 секунд: " << counter << std::endl;  }  // Вывод значения счетчика итераций через 10 секунд  if (counter == 10) {  std::cout << "Значение счетчика через 10 секунд: " << counter << std::endl;  }  // Завершение работы цикла и приложения через 15 секунд  if (counter >= 15) {  break;  }  std::this\_thread::sleep\_for(std::chrono::seconds(1)); // Приостановка выполнения на 1 секунду  }  // Вывод итогового значения счетчика итераций  std::cout << "Итоговое значение счетчика: " << counter << std::endl;  return 0;  } |
| Скриншот запуска |  |

**Задание 3.**

1. Разработайте приложение **OS08\_03,** выполняющее бесконечный цикл.
2. В теле цикла с задержкой подсчитывается количество итераций.
3. Выведите на консоль значения счетчика итераций каждые 3 сек.
4. Корректно завершите работу цикла и приложения через 15 сек., выведите итоговое значение счетчика итераций.

|  |  |
| --- | --- |
| Код программы OS07\_03 | #include <iostream>  #include <chrono>  #include <thread>  int main() {  setlocale(LC\_ALL, "ru");  int counter = 0;  // Функция для выполнения в каждой итерации  auto iterationFunction = [&counter]() {  counter++;  std::cout << "Значение счетчика: " << counter << std::endl;  };  // Создание периодического ожидающего таймера на 3 секунды  std::chrono::milliseconds interval(3000);  std::chrono::steady\_clock::time\_point nextTriggerTime = std::chrono::steady\_clock::now() + interval;  // Бесконечный цикл  while (true) {  // Выполнение функции в каждой итерации  iterationFunction();  // Проверка, прошло ли уже 3 секунды  std::chrono::steady\_clock::time\_point currentTime = std::chrono::steady\_clock::now();  if (currentTime >= nextTriggerTime) {  // Вывод значения счетчика итераций каждые 3 секунды  std::cout << "Значение счетчика через 3 секунды: " << counter << std::endl;  nextTriggerTime += interval;  }  // Завершение работы цикла и приложения через 15 секунд  if (counter >= 15) {  break;  }  std::this\_thread::sleep\_for(std::chrono::milliseconds(500)); // Небольшая пауза для снижения нагрузки на процессор  }  // Вывод итогового значения счетчика итераций  std::cout << "Итоговое значение счетчика: " << counter << std::endl;  return 0;  } |
| Скриншот запуска |  |

**Задание 4.**

1. Разработайте приложение **OS08\_04,** запускающее два одинаковых дочерних процесса **OS08\_04\_X**.
2. Процессы **OS08\_04\_X** вычисляют и выводят на консоль (каждый в свою) пронумерованный ряд простых положительных чисел (простое число делится нацело только на себя и 1).
3. Первый дочерний процесс должен выполняться 1 минуту и корректно завершаться.
4. Первый дочерний процесс должен выполняться 2 минуты и корректно завершаться.
5. Приложение **OS08\_04** завершается после завершения дочерних процессов.

|  |  |
| --- | --- |
| Код программы OS07\_04 | #include <iostream>  #include <windows.h>  #include <tchar.h>  int main() {  setlocale(LC\_ALL, "ru");  STARTUPINFO si;  PROCESS\_INFORMATION pi1, pi2;  ZeroMemory(&si, sizeof(si));  si.cb = sizeof(si);  ZeroMemory(&pi1, sizeof(pi1));  ZeroMemory(&pi2, sizeof(pi2));  if (!CreateProcessW(\_T("..\\x64\\Debug\\OS07\_4\_1.exe"), NULL, NULL, NULL, FALSE, CREATE\_NEW\_CONSOLE, NULL, NULL, &si, &pi1) ||  !CreateProcessW(\_T("..\\x64\\Debug\\OS07\_4\_2.exe"), NULL, NULL, NULL, FALSE, CREATE\_NEW\_CONSOLE, NULL, NULL, &si, &pi2)) {  std::cerr << "Ошибка при создании одного из дочерних процессов." << std::endl;  return 1;  }    WaitForSingleObject(pi1.hProcess, INFINITE);  WaitForSingleObject(pi2.hProcess, INFINITE);  CloseHandle(pi1.hProcess);  CloseHandle(pi1.hThread);  CloseHandle(pi2.hProcess);  CloseHandle(pi2.hThread);  return 0;  } |
| Код программы OS07\_04\_1/2 | #include <iostream>  #include <windows.h>  #include <chrono>  #include <thread>  #include <string>  bool isPrime(int number) {  if (number <= 1) {  return false;  }  for (int i = 2; i <= std::sqrt(number); ++i) {  if (number % i == 0) {  return false;  }  }  return true;  }  void generatePrimes(int duration, int processNumber) {  SetConsoleOutputCP(1251);  int count = 1;  auto startTime = std::chrono::high\_resolution\_clock::now();  while (true) {  auto currentTime = std::chrono::high\_resolution\_clock::now();  auto elapsedSeconds = std::chrono::duration\_cast<std::chrono::seconds>(currentTime - startTime).count();  // Проверяем условие завершения по времени  if (elapsedSeconds >= duration) {  break;  }    if (isPrime(count))  std::cout << "Время: " << elapsedSeconds + 1 << "\tДочерний процесс " << processNumber << ": " << count << std::endl;  else  std::cout << "Время: " << elapsedSeconds + 1 << std::endl;  // Приостанавливаем выполнение на короткое время  std::this\_thread::sleep\_for(std::chrono::milliseconds(500));  count++;  }  }  int main(int argc, char\* argv[]) {  generatePrimes(60, 1);  return 0;  }  #include <iostream>  #include <windows.h>  #include <chrono>  #include <thread>  #include <string>  bool isPrime(int number) {  if (number <= 1) {  return false;  }  for (int i = 2; i <= std::sqrt(number); ++i) {  if (number % i == 0) {  return false;  }  }  return true;  }  void generatePrimes(int duration, int processNumber) {  SetConsoleOutputCP(1251);  int count = 1;  auto startTime = std::chrono::high\_resolution\_clock::now();  while (true) {  auto currentTime = std::chrono::high\_resolution\_clock::now();  auto elapsedSeconds = std::chrono::duration\_cast<std::chrono::seconds>(currentTime - startTime).count();  // Проверяем условие завершения по времени  if (elapsedSeconds >= duration) {  break;  }  if (isPrime(count))  std::cout << "Время: " << elapsedSeconds + 1 << "\tДочерний процесс " << processNumber << ": " << count << std::endl;  else  std::cout << "Время: " << elapsedSeconds + 1 << std::endl;  // Приостанавливаем выполнение на короткое время  std::this\_thread::sleep\_for(std::chrono::milliseconds(500));  count++;  }  // Закрываем консоль дочернего процесса  FreeConsole();  }  int main(int argc, char\* argv[]) {  generatePrimes(120, 2);  return 0;  } |
| Скриншот запуска |  |

**Linux**

**Задание 5.**

1. Разработайте приложение **OS08\_05**.
2. Приложение **OS08\_05** выводит на консоль текущую локальную дату и время в формате ***дд.мм.ггг чч:мин:сек***.

|  |  |
| --- | --- |
| Код программы OS07\_05 | #include <iostream>  #include <ctime>  int main() {  // Получаем текущую локальную дату и время  time\_t currentTime = time(nullptr);  struct tm localTimeInfo;  localtime\_r(&currentTime, &localTimeInfo);  // Форматируем и выводим дату и время  char buffer[80];  strftime(buffer, sizeof(buffer), "%d.%m.%Y %H:%M:%S", &localTimeInfo);  std::cout << "Текущая локальная дата и время: " << buffer << std::endl;  return 0;  } |
| Скриншот запуска |  |

**Задание 6.**

1. Разработайте приложение **OS08\_06,** выполняющее бесконечный цикл.
2. В теле цикла подсчитывается количество итераций.
3. Выведите на консоль значения счетчика итераций через 2 сек. **процессорного** времени и корректно завершите цикл.
4. Приложение **OS08\_06 должно** выполнять замер реального затраченного на работу цикла времени и выводить его значения на консоль.

|  |  |
| --- | --- |
| Код программы OS07\_06 | #include <iostream>  #include <chrono>  #include <unistd.h>  int main() {  using namespace std::chrono;  int iterations = 0;  auto startTime = high\_resolution\_clock::now();  auto startTimeSecondTimer = high\_resolution\_clock::now();  while (true) {  // Замеряем процессорное время каждые 2 секунды  auto currentTime = high\_resolution\_clock::now();  auto elapsedSeconds = duration\_cast<seconds>(currentTime - startTime).count();  auto elapsedSecondsFromStart = duration\_cast<seconds>(currentTime - startTimeSecondTimer).count();  if (elapsedSeconds >= 2) {  std::cout << "Время: " << elapsedSecondsFromStart;  std::cout << "\tЗначение счетчика итераций: " << iterations << std::endl;  startTime = currentTime;  }  iterations++;  // Ожидание небольшого времени для снижения нагрузки процессора  usleep(100000); // Пауза 100 миллисекунд (1 миллион микросекунд = 1 секунда)  }  return 0;  } |
| Скриншот запуска |  |