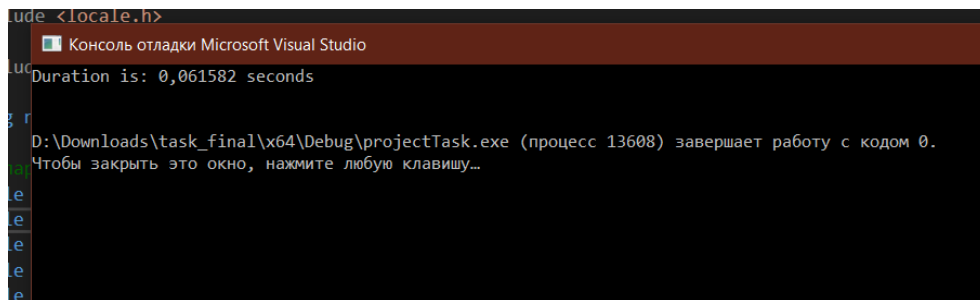


## Отчет по проектному заданию по курсу "Parallel Programming with IPS"

Выполнила Зуева Ирина ИВТ-21М

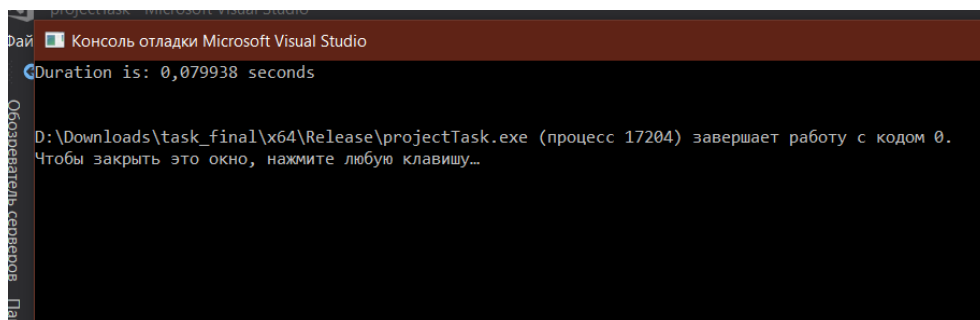
1. Определите время работы последовательной версии разработанной программы в двух режимах: **Debug** и **Release**. Сделайте скрины консоли, где отображается время работы для обоих случаев. Вставьте скрины в отчет к проекту, дав им соответствующие названия.

### Debug



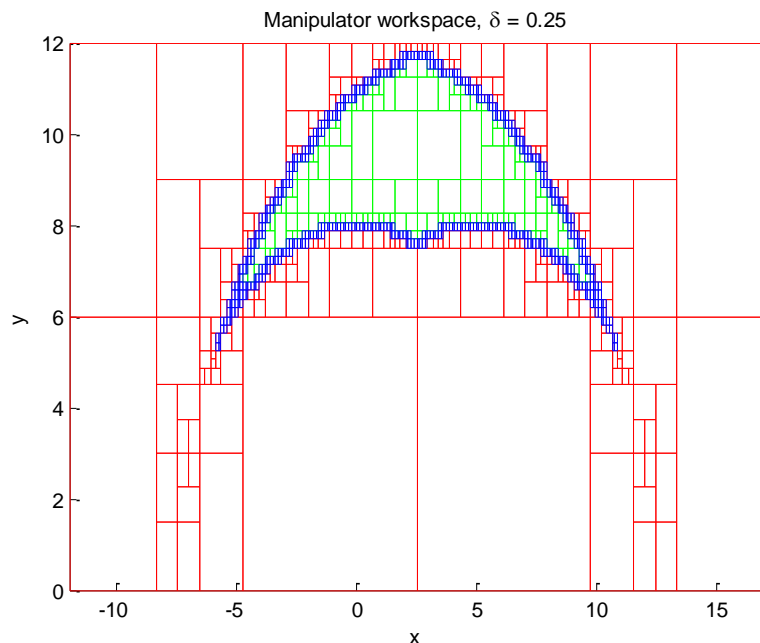
```
ude <locale.h>
Консоль отладки Microsoft Visual Studio
Duration is: 0,061582 seconds
D:\Downloads\task_final\x64\Debug\projectTask.exe (процесс 13608) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...
```

### Release

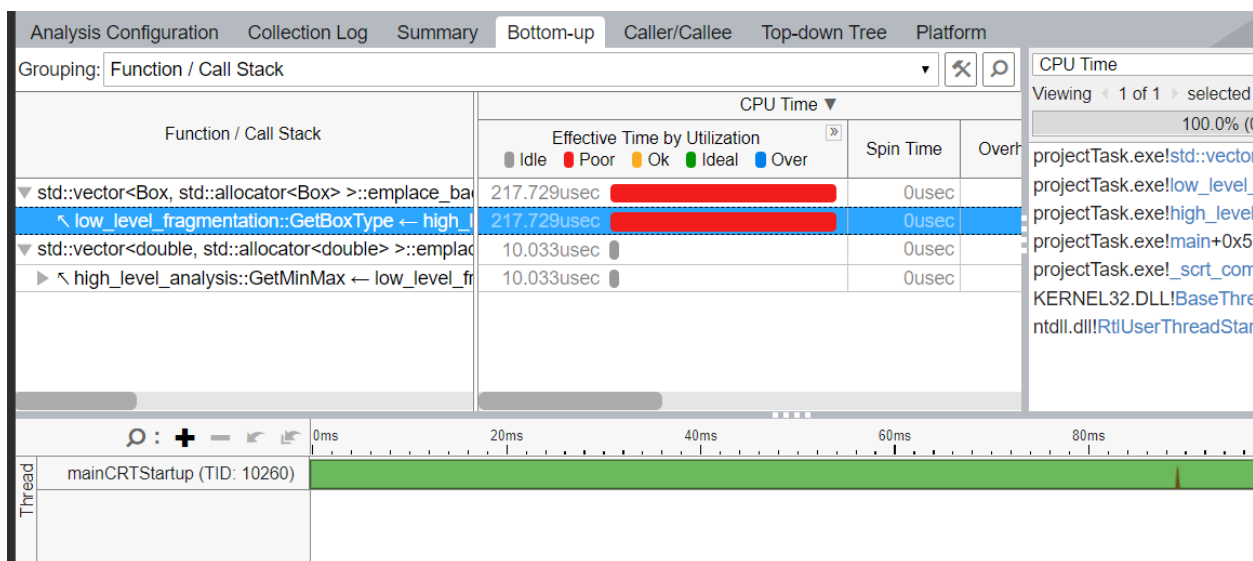


```
Консоль отладки Microsoft Visual Studio
Duration is: 0,079938 seconds
D:\Downloads\task_final\x64\Release\projectTask.exe (процесс 17204) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...
```

2. Постройте полученное рабочее пространство, используя скрипт *MATLAB* PrintWorkspace.m



- Использование **Amplifier XE** в целях определения наиболее часто используемых участков кода. Для этого закомментируйте строки кода, отвечающие за запись результатов в выходные файлы, выберите **New Analysis** из меню **Amplifier XE** на панели инструментов, укажите тип анализа **Basic Hotspots**, запустите анализ. Сделайте скрин окна результатов анализа и вкладки **Bottom-up**. В списке, представленном в разделе **Top Hotspots** вкладки **Summary** должна фигурировать функция **GetMinMax()**.



Hotspots Hotspots by CPU Utilization

Analysis Configuration Collection Log Summary Bottom-up Caller/Callee Top-down Tree Platform

Elapsed Time: 0.100s

CPU Time: 0.000s

Total Thread Count: 1

Paused Time: 0s

Top Hotspots

This section lists the most active functions in your application. Optimizing these hotspot functions typically results in improving overall application performance.

Function	Module	CPU Time
std::vector<Box, std::allocator<Box> >::emplace_back	projectTask.exe	0.000s
std::vector<double, std::allocator<double> >::emplace_back	projectTask.exe	0.000s

\*N/A is applied to non-summable metrics.

4. Использование **Parallel Advisor** с целью определения участков кода, которые требуют наибольшего времени исполнения. Переведите проект в режим **Release** и отключите всякую оптимизацию. Для этого следует выбрать свойства проекта, во вкладке **C/C++** перейти в раздел **Оптимизация**, в пункте меню **“Оптимизация”** выбрать **Отключено (/Od)**. Далее выберем **Parallel Advisor** на панели инструментов **Visual Studio** и запустим **Survey Analysis**. По окончании анализа Вы должны увидеть, что наибольшее время затрачивается в цикле функции **GetSolution()**, двойным кликом по данной строке отчета можно перейти к участку исходного кода и увидеть, что имеется в виду цикл, в котором на каждой итерации вызывается функция **GetBoxType()**. Сделайте скрины результатов **Survey Analysis**, сохраните их, добавьте в отчет. Вернитесь в режим **Debug**.

Intel VTune Amplifier

Elapsed time: 0,16s Vectorized Not Vectorized

FILTER: All Modules All Sources Loops And Functions All Threads

Summary Survey & Roofline Refinement Reports

Some target modules are compiled with inline debug information disabled  
Suggestion: rebuild the modules with the /debug:inline-debug-info option enabled.

Function Call Sites and Loops	Performance Issues	CPU Time	Self Time	Type	Why No Vectorizat
f_sclr_common_main_seh		0,016s	100,0%	0,000s	Function
f_main		0,016s	100,0%	0,000s	Function
[loop in high_level_analysis::GetSolution at Fragmentation.cpp:285]		0,005s	0,000s	0,000s	Scalar
high_level_analysis::GetSolution		0,005s	0,000s	0,000s	Function
[loop in high_level_analysis::GetSolution at Fragmentation.cpp:285]		0,005s	0,000s	0,000s	Scalar

Source Top Down Code Analytics Assembly Recommendations Why No Vectorization?

File: Fragmentation.cpp:285 high\_level\_analysis::GetSolution

Line	Source	Total Time	%	Loop/Function Time	%	Traits
282	{					
283	current_boxes = temporary_boxes;					
284	temporary_boxes = test;					
285	for(int j = 0; j < current_boxes.size(); ++j)			5,031ms		
	Scalar loop					

Selected (Total Time): 0ms

Список ошибок

5. Определение ошибок после введения параллелизации. Запустите анализы **Inspector XE: Memory Error Analysis** и **Threading Error Analysis** на различных уровнях (**Narrowest**, **Medium**, **Widest**). Приложите к отчету скрины результатов запуска перечисленных анализов. Исправьте обнаруженные ошибки, приложите новые скрины результатов анализов, в которых ошибки отсутствуют. *Примечание:* "глюки" **Intel Cilk Plus** исправлять не нужно.

**Detect Leaks**

Target: Analysis Type: Collection Log: Summary

ID	Type	Sources	Modules	Object Size	State
P1	Memory leak	xmemory0	projecttask.exe	43296	New
P2	Memory not deallocated	reducer_vector.h; xmemory0	projecttask.exe	256	New

**Filters**

Severity: Error, Warning  
Type: Memory leak, Memory not deallocated  
Source: reducer\_vector.h, xmemory0

**Code Locations: Memory leak**

Description	Source	Function	Module	Object Size	Variable
Allocation site	xmemory0:880	construct	projecttask.exe	32	block allocated at xmemory0:880

```

878 static void construct(_Alloc, _Objty * const _Ptr,
879 { // construct _Objty(_Types...) at _Ptr
880 ::new (const_cast<void*>(&static_cast<const volatile
881 _Objty(_STD forward<_Types>(_Args)...);
882 }
  
```

**Timeline**

Cilk Worker

**Locate Deadlocks and Data Races**

Target: Analysis Type: Collection Log: Summary

ID	Type	Sources	Modules	State
P1	Data race	list; reducer.h; reducer_vector.h; utility	projecttask.exe	New
P2	Data race	list; vector; xmemory0; utility	projecttask.exe	New
P3	Data race	reducer_vector.h; utility; vector	projecttask.exe	New

**Filters**

Severity: Error  
Type: Data race  
Source: list, reducer.h, reducer\_vector.h, utility

**Code Locations: Data race**

Description	Source	Function	Module	Variable
Read	list:1125	size	projecttask.exe	block allocated at reducer_vector.h:454

```

1123 _NODISCARD size_type size() const noexcept
1124 { // return length of sequence
1125 return (this->Mysize());
1126 }
1127
  
```

**Code Locations: Data race**

Description	Source	Function	Module	Variable
Write	list:1799	_Incsz	projecttask.exe	block allocated at reducer_vector.h:454

```

1797 if (max_size() - this->Mysize() - 1 < _Count)
1798 _Xlength_error("list<T> too long");
1799 this->Mysize() += _Count;
1800 }
  
```

**Timeline**

main (17356)  
Cilk Worker (16616)  
Cilk Worker (7528)

6. Изменяя  $X$ , запускайте программу и фиксируйте время ее выполнения, каждый раз сохраняйте скрины консоли, где должно быть отображено количество вычислителей (`cout << "Number of workers " << __cilkrts_get_nworkers() << endl;`) и время работы программы.

```
et
Консоль отладки Microsoft Visual Studio
Duration is: 0,098959 seconds
ig
Number of workers 1
c
ai
D:\Downloads\task_final\x64\Debug\projectTask.exe (процесс 2896) завершает работу с кодом 0.
/
Чтобы закрыть это окно, нажмите любую клавишу...
or
ig
ri
ig
```

```
je
Консоль отладки Microsoft Visual Studio
Duration is: 0,234435 seconds
1
Number of workers 2
2
2
D:\Downloads\task_final\x64\Debug\projectTask.exe (процесс 3468) завершает работу с кодом 0.
2
Чтобы закрыть это окно, нажмите любую клавишу...
2
2
2
2
```

```
t main()
Консоль отладки Microsoft Visual Studio
Duration is: 0,078664 seconds
Number of workers 3
D:\Downloads\task_final\x64\Debug\projectTask.exe (процесс 16892) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...
```

```
Правка Вид Проект Сборка Отладка Команда Средства Тест Анализ Окно Справка
Консоль отладки Microsoft Visual Studio
00
Duration is: 0,148081 seconds
Number of workers 4
D:\Downloads\task_final\x64\Debug\projectTask.exe (процесс 6348) завершает работу с кодом 0.
Чтобы закрыть это окно, нажмите любую клавишу...
```

7. Визуализация полученного решения. Поэкспериментируйте со входными параметрами программы и отобразите несколько версий полученного рабочего пространства робота. Рисунки приложите к отчету.

