МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра вычислительной техники

Отчет по лабораторной работе №2
по дисциплине «Операционные системы»

Тема: Управление памятью

Студент гр. 9308 Дубенков С.А

Преподаватель Тимофеев А.В.

Санкт-Петербург

2021

Цель работы

Исследовать механизмы управления виртуальной памятью Win32.

Код обоих заданий представлен на Github:
https://github.com/sedub01/eltech/tree/main/OS/2

Задание 2.1

Было создано приложение, которое:

- получает информацию о вычислительной системе
- определяет статус виртуальной памяти
- определяет состояние конкретного участка памяти по заданному с клавиатуры адресу
- резервирует регион в автоматическом режиме и в режиме ввода адреса начала региона
- резервирует регион и передаёт ему физическую память в автоматическом режиме и в режиме ввода адреса начала региона
- записывает данные в ячейки памяти по заданным с клавиатуры адресам
- устанавливает защиту доступа для заданного (с клавиатуры) региона памяти и проверяет ее
- возвращает физическую память и освобождает регион адресного пространства, заданного (с клавиатуры) региона памяти

Главное меню

```
What do you wanna choose?
1 - get info about computing system
2 - define status of virtual memory
3 - define the state of memory area by the address adress set from keyboard
4 - reservation of the region in automatic mode and in the mode
    enter the address of the beginning of the region
5 - reservation of the region and transfer of physical memory to it in automatic
    mode and in the mode of entering the address of the beginning of the region
6 - writing data to memory cells at addresses specified from the keyboard
7 - setting access protection for a specified (from the keyboard) memory
    region and checking it
8 - returning physical memory and freeing the region of the address
    space of the specified (from the keyboard) region of memory
0 - exit
Enter your choice:
```

Демонстрация результатов:

- получение информации о вычислительной системе

```
x86 architechture
The page size: 4096
The lowest memory address accessible: 0x10000
The highest memory address accessible: 0x7ffeffff
A mask representing the set of processors
configured into the system:
00000000000000000000111111111111
The number of logical processors in the current group: 12
The granularity for the starting address
at which virtual memory can be allocated: 65536
The architecture-dependent processor level: 23
The architecture-dependent processor revision: 24577
```

- определение статуса виртуальной памяти

```
The size of the MEMORYSTATUS data structure, in bytes: 32
The approximate percentage of physical memory that is in use: 41
The amount of actual physical memory, in bytes: 2147483647
The amount of physical memory currently available, in bytes: 2147483647
The current size of the committed memory limit, in bytes: 4294967295
The maximum amount of memory the current process can commit, in bytes: 4294967295
The size of the user-mode portion of the virtual address space
of the calling process, in bytes: 2147352576
The amount of unreserved and uncommitted memory currently in
the user-mode portion of the virtual address
space of the calling process, in bytes: 2128121856
```

- определение состояния конкретного участка памяти по заданному с клавиатуры адресу

```
Enter the required valid address: 0x1200fc
A pointer to the base address of the region of pages to be queried: 0x1200fc
A pointer to the base address of the region of pages: 0x120000
A pointer to the base address of a range of pages allocated by the VirtualAlloc function: 0xc0000
The memory protection option when the region was initially allocated: 2
The size of the region beginning at the base address
in which all pages have identical attributes, in bytes: 430080
Indicates committed pages for which physical
storage has been allocated, either in memory
or in the paging file on disk
The access protection of the pages in the region: 2
PAGE_READONLY

The type of pages in the region
Indicates that the memory pages within
the region are mapped into the view of a section
```

- резервирование региона в автоматическом режиме и в режиме ввода адреса начала региона

```
What do you wanna choose?
1 - reserve the region in automatic mode
2 - reserve the region in the mode enter the address of the beginning of the region
Enter your choice: 1

Memory area allocated
Address: 0x1e0000
```

Выбор второго пункта меню

```
Enter the required valid address: 0x7dcf32
Memory area allocated
Address: 0x7d0000
```

- резервирование региона и передача ему физической памяти в автоматическом режиме и в режиме ввода адреса начала региона

```
What do you wanna choose?
1 - reserve the region in automatic mode
2 - reserve the region in the mode enter the address of the beginning of the region
Enter your choice: 1

Memory area allocated
Address: 0x1f0000
```

Выбор второго пункта меню

```
Enter the required valid address: 0x77fdc3
Memory area allocated
Address: 0x770000
```

- запись данных в ячейки памяти по заданным с клавиатуры адресам

```
Enter data for input: eqeqw
Enter the address of input: 0x123456
Access is denied
Enter data for input: ererfsdf dsfs
Enter the address of input: 0x100000
Memory area 0x186a0 filled. Entered data: ererfsdf dsfs
```

- установка защиты доступа для заданного (с клавиатуры) региона памяти и ее проверку

```
Enter the address: 0x100000
Choose new protection level:
1 - PAGE_EXECUTE
2 - PAGE_EXECUTE_READ
3 - PAGE_EXECUTE_READWRITE
4 - PAGE_EXECUTE_WRITECOPY
5 - PAGE_NOACCESS
6 - PAGE_READONLY
7 - PAGE_READWRITE
8 - PAGE_WRITECOPY
9 - PAGE_TARGETS_INVALID
10- PAGE_TARGETS_NO_UPDATE
Enter your choice: 6

Old protection level:
PAGE_READWRITE
Press any key to continue . . .
```

Перепроверка:

```
Enter the address: 0x100000
Choose new protection level:
1 - PAGE_EXECUTE
2 - PAGE_EXECUTE_READ
3 - PAGE_EXECUTE_READWRITE
4 - PAGE_EXECUTE_WRITECOPY
5 - PAGE_NOACCESS
6 - PAGE_READONLY
7 - PAGE_READWRITE
8 - PAGE_WRITECOPY
9 - PAGE_TARGETS_INVALID
10- PAGE_TARGETS_NO_UPDATE
Enter your choice: 7
Old protection level:
PAGE_READONLY
Press any key to continue . . .
```

- возврат физической памяти и освобождение региона адресного пространства, заданного (с клавиатуры) региона памяти
  Сначала надо зарезервировать адрес памяти (сделано в п.5)

```
Address: 0xd0000
```

Теперь эту память можно освободить

```
Enter the address: 0xd0000
Memory area's released
Для продолжения нажмите любую клавишу . . .
```

Вывод: в ходе выполнения программы была проверена корректность её работы. Резервирование и освобождение для регионов памяти выполняется правильно, за исключением ситуаций, когда регион уже зарезервирован ранее. Успешность изменения уровня доступа памяти была проверена - изменения корректно отображены.

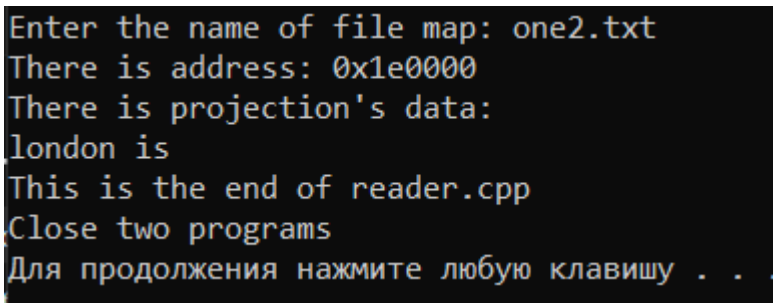
Автоматические режимы резервирования регионов памяти и передачи им физ. памяти были реализованы при помощи передачи в функцию VirtualAlloc() параметра NULL в качестве начального адреса региона для выделения. При резервировании памяти с заданием начального адреса важно учитывать округление исходного адреса до ближайшей степени гранулярности выделения памяти (в нашем случае 4096 бит).

Задание 2.2

Для проверки работоспособности механизма нужно запустить программу-писатель, ввести имя файла, затем его отображения, а после - данные, которые необходимо записать

```
Enter file name (in English, plez): one.txt
Enter file mapping name: one2.txt
File projected successfully.
Projection's address: 0xd0000
Enter data for input:
> london is
Data entered. Don't close writer-file
Для продолжения нажмите любую клавишу . . .
```

После чего, не закрывая программу-писатель, нужно запустить файл-читатель и вести файл отображения



Вывод: судя по работе двух программ механизм работает отлично. Обе программы ссылаются на одно и то же отображение. Видно, что адреса проекций в программах различаются. Происходит это потому, что два процесса могут совместно использовать объект «проецируемый файл», при этом, при помощи функции MapViewOfFile() каждый процесс проецирует этот объект на свое ВАП и используют эту часть адресного пространства как разделяемую область данных.