

Библиотека для написания программ, запускаемых при помощи GRUB

ВЫПОЛНИЛ СТУДЕНТ ГР. ИУ9-52

А.Б. БАРЛУКА

РУКОВОДИТЕЛЬ КУРСОВОЙ РАБОТЫ

А.В. КОНОВАЛОВ

Цели и задачи работы

Целью курсовой работы является создание инструмента для написания на высокоуровневом языке программ, работающих вне операционной системы;

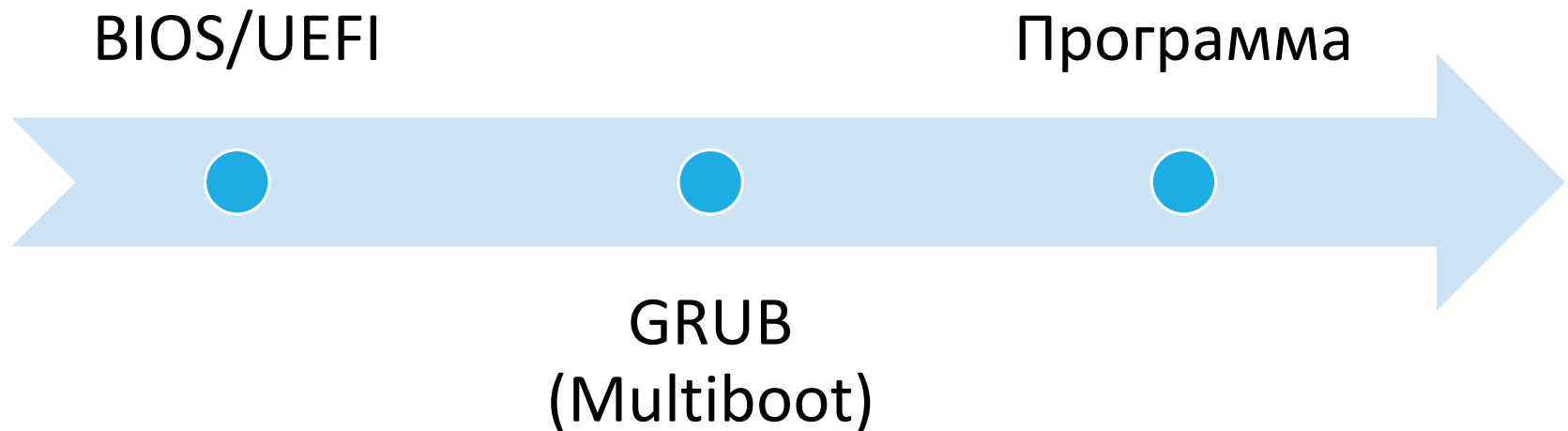
Достижение указанной цели осуществлялось путем решения следующих основных задач:

- 1) Определение структуры программ;
- 2) Сборка программы;
- 3) Реализация библиотек

Процесс загрузки ОС



Процесс загрузки программы



Пример программы

В данном примере в бесконечном цикле считывается строка, а затем выводится на экран:

```
#include "screen.h"
#include "types.h"
#include "cursor.h"

int cursor_x;
int cursor_y;

void main()
{
    clear_screen();
    char string[256];
    for (;;) {
        gets(string);
        putchar('\n');
        puts(string);
        putchar('\n');
    }
}
```

Структура disk.img

- grub
- kernel.bin

```
grub@grub-VirtualBox:/media/grub/483539be-a19e-4b0a-b5b8-b1c00a847da8$ tree -C -L 3
.
├── boot
│   └── grub
│       ├── fonts
│       ├── grub.cfg
│       ├── grubenv
│       ├── i386-pc
│       └── locale
├── kernel.bin
└── lost+found [error opening dir]

6 directories, 3 files
```

Компиляция kernel.bin

- ❑ Утилита Make
- ❑ GCC (*.c)
- ❑ GNU Assembler (loader.s)
- ❑ Компоновщик ld (linker.ld)



Работа с периферийными устройствами

❑ Командные порты

- Video Graphics Array, VGA (0x3D4, 0x3D5)
- Клавиатура (0x60, 0x64)
- Programmable Interval Timer, PIT (0x40, 0x43)
- Real-Time Clock, RTC (0x70, 0x71)

```
unsigned char inb(unsigned short int port);  
void outb(unsigned char value, unsigned short int port);
```

❑ Буфер видеопамати VGA по адресу 0xb8000

Основные функции библиотеки

❑ Вывод текста на экран

```
void clear_screen();  
void putchar(int c);  
int puts(const char* s);  
int printf(const char *format, ...);
```

❑ Чтение с клавиатуры

```
int getchar();  
char* gets(char* s);
```

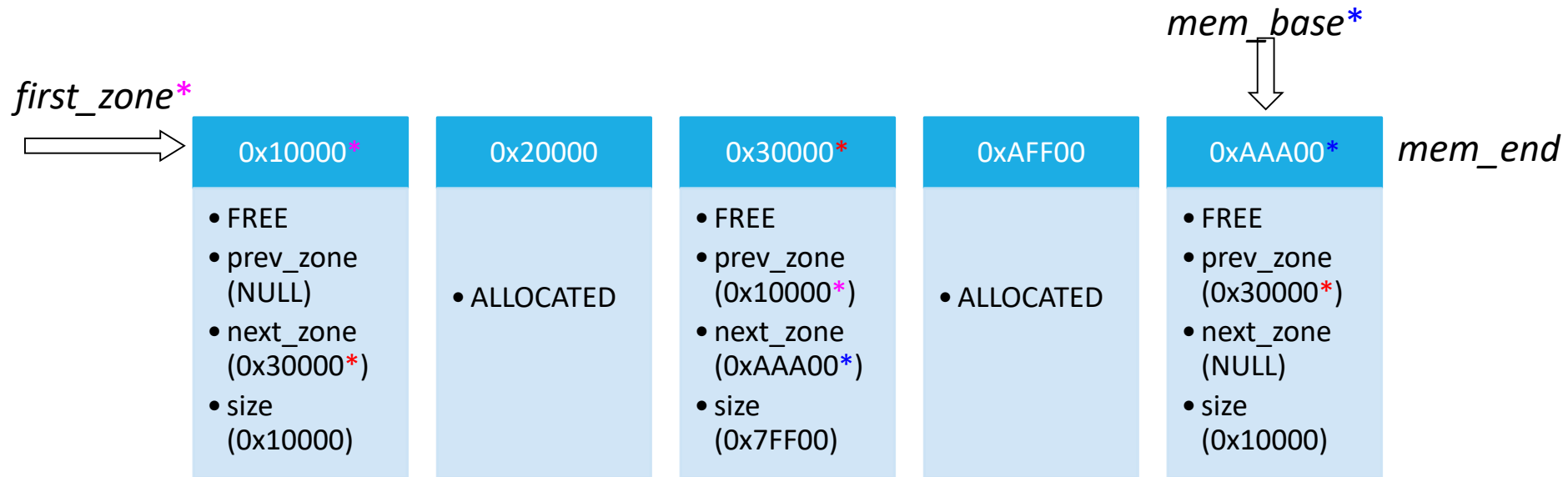
❑ Работа с динамической памятью

```
void mem_init(multiboot_info_t* mbd);  
void* malloc(size_t size);  
void free(void* ptr, size_t size);
```

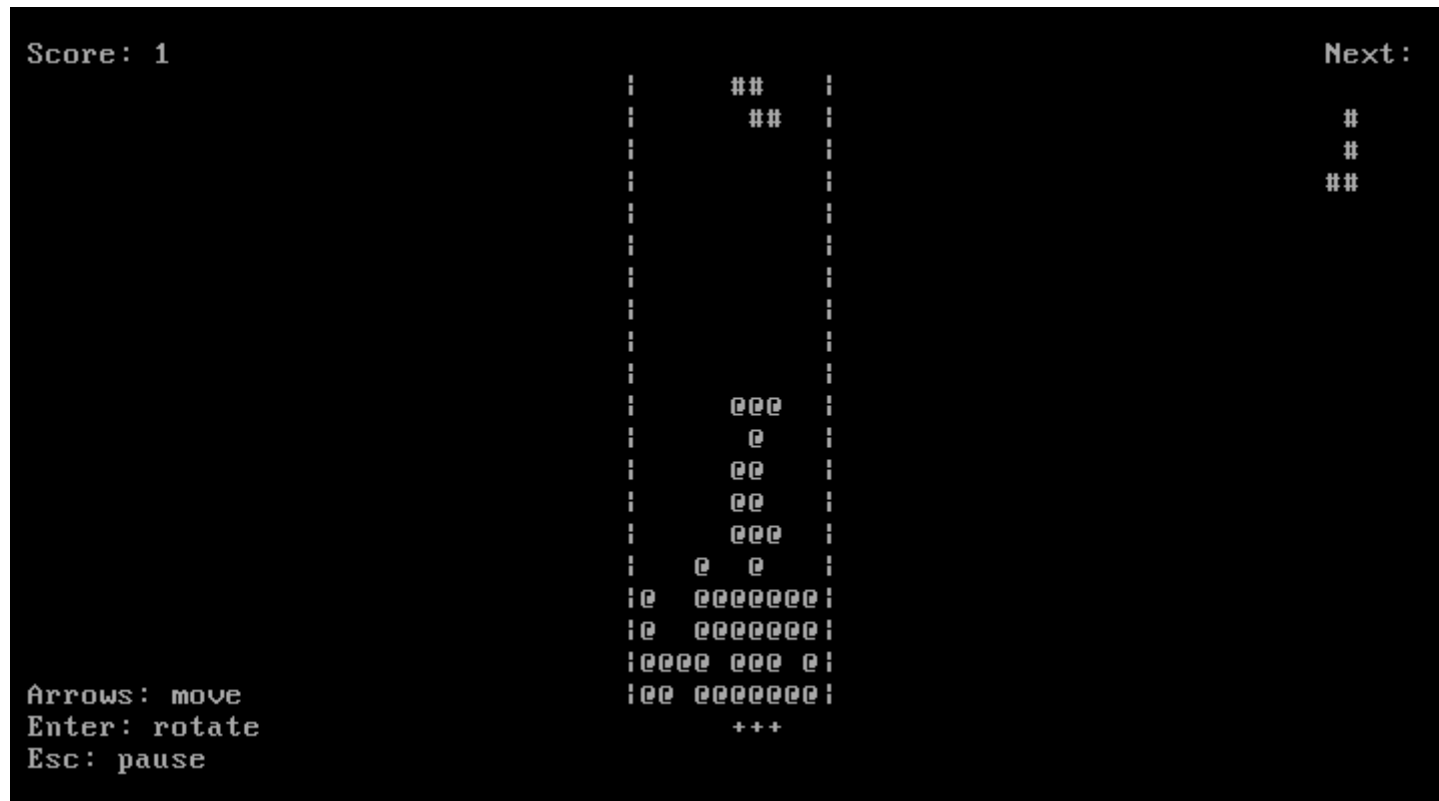
❑ Приостановка выполнения

```
void delay(unsigned int ticks);  
void sleeps(unsigned int seconds);
```

Алгоритм работы с памятью



Демонстрационное приложение: Тетрис



Выводы

- ❖ Изучена обширная теоретическая база по созданию операционных систем;
- ❖ Была разработана автоматизированная система сборки программ;
- ❖ Реализованы библиотеки ввода-вывода и работы с динамической памятью.