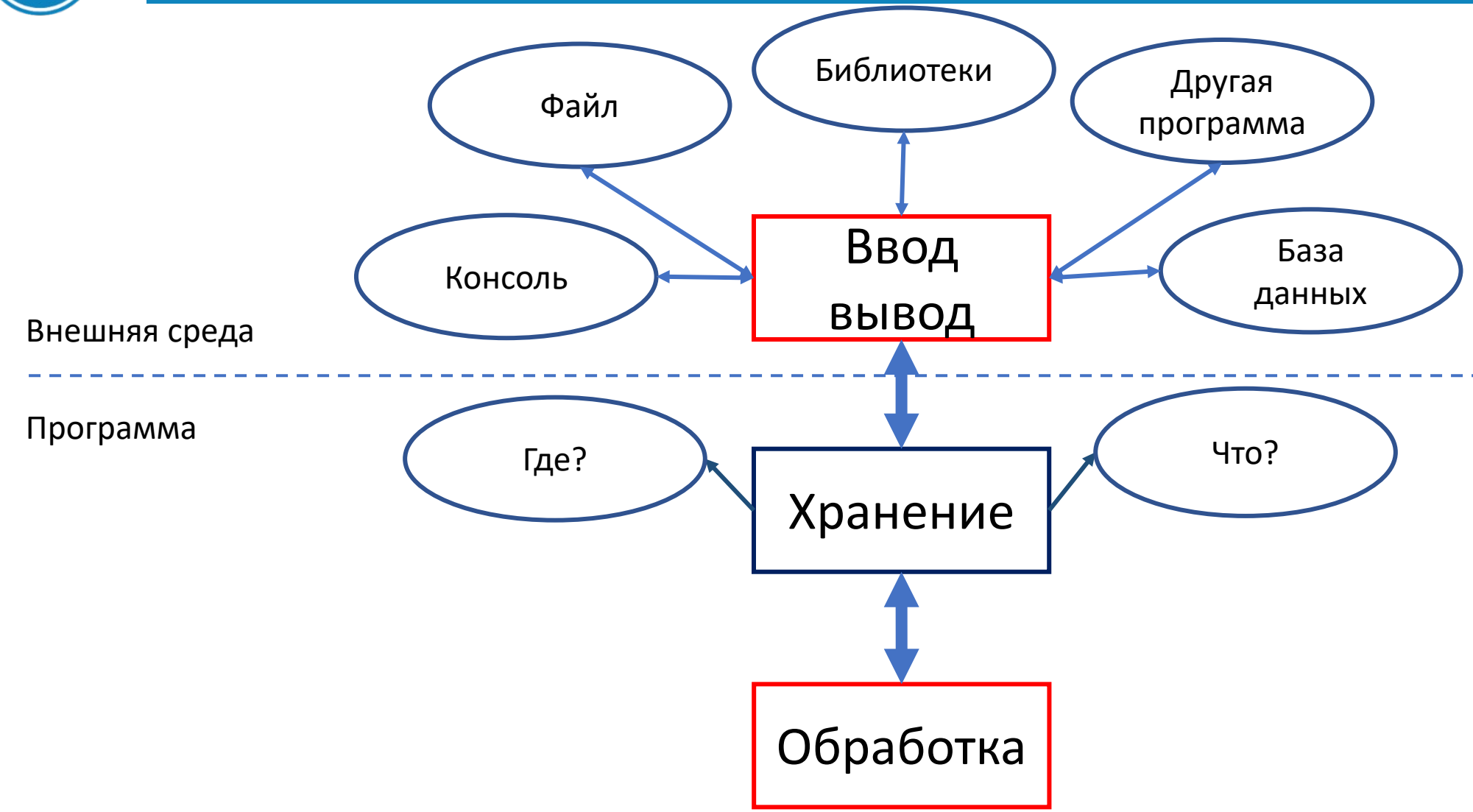
Основы программирования на C++

Занятие 7. Строки, структуры

Дерево языка



Проблема

Необходимо разработать программу для хранения и обработки домашней библиотеки.

Вопросы

1. Как хранить названия книг?
2. Как лучше хранить список книг?
3. Как сохранять список книг?

Строки в Си

Строк в Си НЕТ

Строки в Си

Строки == Массив символов

```c
#include <stdio.h>

const int N = 10;

int main()
{
    char name[N];
    printf("Enter name: ");
    scanf_s("%s", name, N);
    printf("Your name is %s.", name);
    return 0;
}
```

```
Enter name: SAB
Your name is SAB.
```

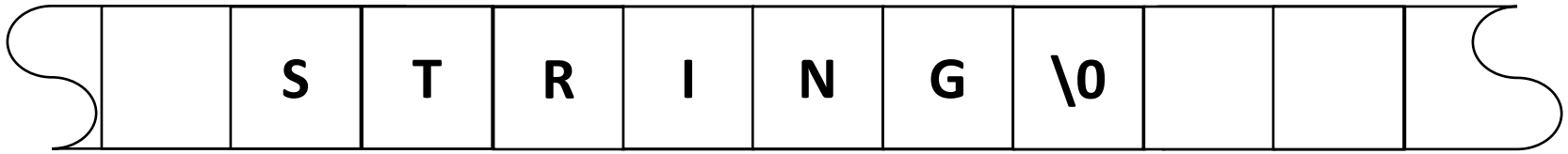
Строки в Си

```c
#include <stdio.h>
#include <string.h>

const int N = 10;

int main()
{
    char name[N];
    printf("Enter string: ");
    scanf_s("%s", name, N);
    printf("String is %s.\n", name);
    printf("String length is %zu\n", strlen(name));
    printf("\nSymbol\tCode\n");
    for (int i = 0; i < N; i++)
        printf("%c\t0x%x\n", name[i], name[i]);
    return 0;
}
```

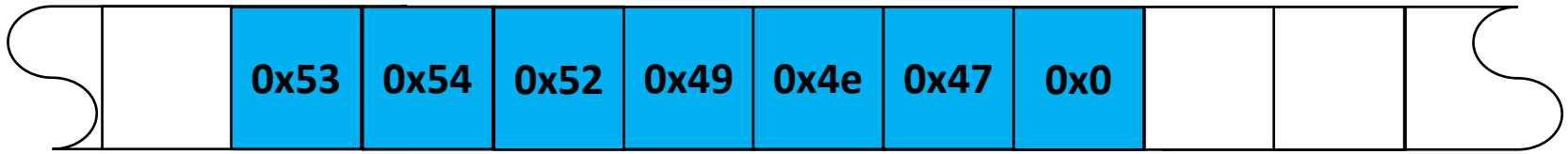
```
Enter string: SAB
String is SAB.
String length is 3

Symbol  Code
S       0x53
A       0x41
B       0x42
        0x0
■       0xfffffffe
■       0xfffffffe
■       0xfffffffe
■       0xfffffffe
■       0xfffffffe
■       0xfffffffe
■       0xfffffffe
```

Как хранится строка в памяти

Символы



Коды символов



Полезные функции

```
#include <string.h>
```

	Синтаксис	Описание
1	size_t *strlen (const char *str);	Функция strlen вычисляет количество символов в строке до первого вхождения символа конца строки.
2	char *strcpy (char *destination, const char *source);	Функция strcpy копирует данные из строки, на которую указывает аргумент source, в строку, на которую указывает аргумент destination, пока не встретится символ конца строки (нулевой символ).
3	int strcmp (const char *str1, const char *str2);	Функция сравнения строк. Возвращает 0 – если сравниваемые строки идентичны.

Сравнение строк

```c
#include <stdio.h>
#include <string.h>

const int N = 10;

int main()
{
    char str1[N];
    char str2[N];
    scanf_s("%s", str1, N);
    scanf_s("%s", str2, N);
    printf("String is %s\n", str1);
    printf("String is %s\n", str2);
    int res = strcmp(str1, str2); // strings compare
    printf("Res = %d\n", res);
    return 0;
}
```

str1 < str2 res = -1

str1 > str2 res = 1

str1 = str2 res = 0

Копирование строк

```c
#include <stdio.h>
#include <string.h>

const int N = 10;

int main()
{
    char str1[N];
    char str2[N];
    scanf_s("%s", str1, N);
    printf("String 1 is %s\n", str1);
    strcpy_s(str2, str1); // strings copy str1 to str2
    printf("String 2 is %s\n", str2);
    return 0;
}
```

Структуры

```
const int N = 10;
const int N_students = 2;

char name[N_students][N];
char surname[N_students][N];
int grade[N_students];

for (int i = 0; i < N_students; ++i){
    printf("Name: ");
    scanf_s("%s", name[i], N);
    printf("Surname: ");
    scanf_s("%s", surname[i], N);
    printf("Grade: ");
    scanf_s("%d", &grade[i]);
}
for (int i = 0; i < N_students; ++i){
    printf("Name = %s\n", name[i]);
    printf("Surname = %s\n", surname[i]);
    printf("Grade = %d\n", grade[i]);
}
```

Как лучше хранить список объектов состоящие из нескольких частей?

Полный код см. в заметках к слайду

Структуры

```
struct Student
{
    char name[N];
    char surname[N];
    int grade;
};
```

Структура

Поля структуры

```
struct Student student[N_students];
```

Объявление

```
student[i].name
```

Обращение к полю

Полный код см. в заметках к слайду

Выделение памяти под структуру

```c
struct personT p1, * p2;

   // need to malloc space for the name field:
   p1.name = (char*)malloc(sizeof(char) * 8);
   strcpy(p1.name, "Zhichen");
   p1.age = 22;


   // first malloc space for the struct:
   p2 = (struct personT*)malloc(sizeof(struct personT));

   // then malloc space for the name field:
   p2->name = (char*)malloc(sizeof(char) * 4);
   strcpy(p2->name, "Vic");
   p2->age = 19;
```