Переписанные на LaTeXвопросы для экзамена по информатике

Морозов Н.Ю

5 декабря 2023 г.

Содержание

1 Базовые конструкции программирования: синтаксис и семантика языков высокого уровня; переменные, типы, выражения и присваивания; простейший ввод/вывод; 3

2 Условные предложения и итеративные конструкции 5

3 Функции и передача параметров; структурная декомпозиция. 7

1 Базовые конструкции программирования: синтаксис и семантика языков высокого уровня; переменные, типы, выражения и присваивания; простейший ввод/вывод;

Языки программирования – это формальные искусственные языки. Как и естественные языки, они имеют алфавит, словарный запас, грамматику и синтаксис, а также семантику.

Алфавит – разрешенный к использованию набор символов, с помощью которого могут быть образованы слова и величины данного языка.

Синтаксис – система правил, определяющих допустимые конструкции языка программирования из букв алфавита.

Семантика – система правил однозначного толкования каждой языковой конструкции, позволяющих производить процесс обработки данных.

Взаимодействие синтаксических и семантических правил определяет основные понятия языка, такие как операторы, идентификаторы, константы, переменные, функции, процедуры и т.д. В отличие от естественных, язык программирования имеет ограниченный запас слов (операторов) и строгие правила их написания, а правила грамматики и семантики, как и для любого формального языка, явно однозначно и четко сформулированы.

Языки программирования, ориентированные на команды процессора и учитывающие его особенности, называют языками низкого уровня. «Низкий уровень» не означает неразвитый, имеется в виду, что операторы этого языка близки к машинному коду и ориентированы на конкретные команды процессора.

Языком самого низкого уровня является ассемблер. Программа, написанная на нем, представляет последовательность команд машинных кодов, но записанных с помощью символьных мнемоник. В таком случае программист получает доступ ко всем возможностям процессора. Однако, при этом требуется хорошо понимать устройство компьютера, а использование такой программы на компьютере с процессором другого типа невозможно. Такие языки программирования используются для написания небольших системных приложений, драйверов устройств, модулей стыковки с нестандартным оборудованием.

Языки программирования, имитирующие естественные, обладающие укрупненными командами, ориентированные «на человека», называют языками высокого уровня. Чем выше уровень языка, тем ближе структуры данных и конструкции, использующиеся в программе, к понятиям исходной задачи. Особенности конкретных компьютерных архитектур в них не учитываются, поэтому исходные тексты программ легко переносимы на другие платформы, имеющие трансляторы этого языка. Разрабатывать программы на языках высокого уровня с помощью понятных и мощных команд значительно проще, число ошибок, допускаемых в процессе программирования, намного меньше. В настоящее время насчитывается несколько сотен таких языков (без учета их диалектов).

Программы оперируют с различными данными, которые могут быть простыми и структурированными. Простые данные - это целые и вещественные числа, символы и указатели (адреса объектов в памяти). Структурированные данные - это массивы и структуры;

Переменные - это именованные области памяти, используемые для хранения данных. Каждая переменная имеет свой тип данных, который определяет, какой вид информации может быть сохранен в данной переменной.

В языке различают понятия "тип данных" и "модификатор типа". **Тип данных** - это, например, целый, а модификатор - со знаком или без знака. Целое со знаком будет иметь как положительные, так и отрицательные значения, а целое без знака - только положительные значения. В языке Си можно выделить пять базовых типов, которые задаются следующими ключевыми словами:

- char - символьный;
- int - целый;
- float - вещественный;
- double - вещественный двойной точности;
- void - не имеющий значения;

Дадим им краткую характеристику:

Дадим им краткую характеристику:

1. Переменная типа char имеет размер 1 байт, её значениями являются символы.
2. Размер переменной типа int в стандарте языка Си не определён. В болшинстве систем программирования размер переменной типа int соотвествует размеру целого машинного слова. В этом случае знаковые значения этой переменной могут лежать в диапазоне от -32768 до 32767.

3. Ключевое слово float позвляет определить переменные вещесвенного типа. Их значения имеют дробную часть, отделяемую точкой, например: $-5.6, 31.28$ и т.п. Вещественные числа могут быть записаны также в форме с плавающей точкой, например: $-1.09e+4$. Число перед символом "e"называется мантиссой, а после "e порядком. Она может принимать значения в диапазоне от $3.4e-38$ до $1.7e+308$.
4. Ключевое слово double позвляет определить вещественную переменную двойной точности. Она занимет в памяти в два раза больше места, чем переменная типа float. Переменная типа double может принимать значения в диапазоне от $1.7e-308$ до $1.7e+308$.
5. Ключевое слово void (не имеющий значения) используется для нейтрализации значения объектов, например, для объявления функции, не вовзращающей никаких значений.

Объект некоторого базового типа может быть модифицирован. С этой целью используются специальные ключевые слова, называемые модификаторами. В стандарте ANSI языка Си имеются следующие модификаторы типа:

- unsigned
- signed
- short
- long

Модификаторы записываются перед спецификаторами типа, например: unsigned char. Если после модификатора опущен спецификатор, то компилятор предполагает, что этим спецификатором является int. Таким образом следующие строки:

```
long a;
long int a;
```

Все переменные до их использования должны быть определены (объявлены). При этом задается тип, а затем идет список из одной или более переменных этого типа, разделенных запятыми. Например:

```
int a, b, c;
char x, y;
```

Переменные в языке Си могут быть инициализированы при их определении:

```
int a = 25, h = 6;
char g = 'Q', k = 'm';
float r = 1.89;
long double n = r*123;
```

В языке возможны глобальные и локальные объекты. Первые определяются вне функций и, следовательно, доступны для любой из них. Локальные объекты по отношению к функциям являются внутренними. Они начинают существовать, при входе в функцию и уничтожаются после выходы из неё.

```
int a;              /* Объявление глобальной переменной */

int function(int b, char c);   /* Объявление функции (т.е. описание её заголовка) */

void main(void)

{                   //Тело программы
    int d, e;       //Определение локальных переменных
    float f;        //Определение локальной перемной
    ...
}
```

Операции ввода/вывода в языке Си организованы посредством библиотечных функций.
Самый простой механизм ввода - чтение по одному символу из стандартного входного потока с помощью функции getchar():

```
x = getchar();
```

присваевает переменной x очередной вводимый символ. Переменная x должна иметь символьный или целый тип.
Другая функция - putchar(x) выдаёт значение переменной x в стандартный выходной поток:

int putchar(int);

Объявления getchar() и putchar() сделаны в заголовочном файле stdio.h содержащем описания заголовков библиотечных функций стандартного ввода/вывода. Подключение осуществляется с помощью директивы препроцессора:

include <stdio.h>

Функция printf() обеспечивает форматированный вывод. Ее можно записать в следующем формальном виде:

printf("управляющая срока" , аргумент_1, аргумент_2);

Управляющая строка содержит компоненты трех типов: обычные символы, которые просто копируются в стартный выходной поток (выводятся на экран дисплея); спецификации преобразования, каждая из которых вызывает вывод на экран очередного аргумента из последующего списка; управляющие символьные константы.

Каждая спецификация преобразования начинается со знака % и заканчивается некотороым символом, задающим преобразование. Между знаком % и символом преобразования могут встречаться другие знаки в соотвествии со следующим форматом:

На место символа преобразования могут быть записаны:

c – значением аргумента является символ;
d или **i** – значением аргумента является десятичное число;
e – значением аргумента является вещественное десятичное число в экспоненциальной форме вида $1.23e+2$;
f – значением аргумента является вещественное десятичное число с плавающей точкой;
s – значением аргумента является строка символов (символы строки выводятся до тех пор, пока не встретится символ конца строки или же не будет, выведенно число символов, заданное точностью);
u – значением аргумента является беззнаковое целое число;
p – значением аргумента является указатель;
n – применятся в операциях форматирования. Аргумент, соответсвтующий этому символу спецификации, должен быть указателем на целое. В него возвращается номер позиции строки (отображаемой на зкране), в которой записана спецификация %n.

Функция printf() использует управляющую строку, чтобы определить, сколько всего аргументов и каковы их типы. Аргументами могут быть переменные, константы, выражения, вызовы функций; главное, чтобы их значения соотвествовали заданной спецификации.

Функция scanf() обеспечивает форматированный ввод. Ее можно записать в следующим виде:

scanf("управляющая строка" , аргумент_1, аргумент_2, ...);

Аргументы scanf() должны быть указателями на соответствующие значения. Для этого перед именем переменной записывается символ & – оперция получения адреса в памяти.

Управляющая строка содержит спецификации преобразования и используется для установдения количества и типов аргументов

2 Условные предложения и итеративные конструкции

Операторы цикла

Циклы организуются, чтобы выполнить некоторый оператор или группу операторов определенное число раз. В языке Си три оператора цикла: for, while, do - while. Первый из них формально записывается, в следующем виде:

for (выражение_1; выражение_2; выражение_3) тело_цикла

Тело цикла составляет либо один оператор, либо несколько операторов, заключенных в фигурные скобки { ... } (после блока точка с запятой не ставится). В выражениях 1, 2, 3 фигурирует специальная переменная, называемая управляющей. По её значению устанавливается необходимость повторения цикла или выхода из него.

Выражение_1 присваевает начальное значение управляющей перменной, выражение_3 изменяет его на каждом шаге, а выражение_2 проверяет, не достигло ли оно граничного значения, установляющего необходимость выхода из цикла.

Примеры:

```
for (i = 1; i < 10; i++)
{ ...
}

for (ch = 'a'; ch != 'p';) scanf("%c", &ch);
```

/* Цикл будет выполняться до тех пор, пока с калвиатуры
не будет символ символ 'p' */

Любое из трёх выражений в цикле for может отсутствовать, однако точка с запятой должна оставаться. Таким образом, for (;;) {...} – это бесконечный цикл, из которого можно выйти лишь другими способами. Допускается вложенные конструкции, т.е. в теле некоторого цикла могут встречаться другие оперторы for.

Оператор while формально записывается в таком виде:

while(выражение) тело_цикла

Выражение в скобках может принимать ненулевое (истинное) или нулевое (ложное) значение. Если оно истинно, то выплнятеся тело цикла и выражение вычисляется снова. Если выражение ложно, то цикл while заканчивается.

Оператор do-while формально записывается следующим образом:

do {тело_цикла} while(выражение);

Основным отличием между циклами while и do - while является то, что тело в цикле do - while выполняется по крайней мере один раз. Тело цикла будет выполняться до тех пор, пока выражение в скобках не примет ложное значеие. Если оно ложно при входе в цикл, то его тело выполняется ровно один раз. Допускается вложенность одних циклов в другие, т.е. в теле любого цикла могут появляться операторы for, while и do - while.

В теле цикле могут использоваться новые операторы break обеспечивает немедленный выход из цикла, оператор continue вызывает прекращение очередной и начало следеующей итерации.

Операторы условных и безусловных переходов

Для организации условных и безусловных переходов в программе на языке Си используются операторы: if - else switch и goto. Перевый из них записывается следующим образом:

if(проверка_условия)оператор_1; else оператор_2;

Если условие в скобках принимает истинное значение, выполняется оператор_1, если ложное - оператор_2. Если вместо одного необходимость выполнить несколько операторов, то они заключаются в фигурные скобки. В операторе if слово else может отсутствовать.

В операторе if - else непосредсвенно после ключевых слов if и else должны следовать другие операторы. Если хотя бы один из них является оператором if, его называют вложенным. Солгласно принятому в языке Си соглашению слово elst всегда относится к ближайшему предшествующему ему if.

Оператор switch позволяет выбрать одну из нескольких албтернатив. Он записывается в следующем формальном виде:

```
switch(выражение)
{
    case константа_1: операторы_1;
         break;

    case константа_2: операторы_2;
         break;
    ...       ...
    defaults: операторы_defaults;
}
```

Здесь вычисляется значение целого выражения в скобках (его иногда называют селектором) и оно сравнивается со всеми константами (константными выражениями). Все константы должны быть различными. При совпадении выполнится соответсвующий вариант операторов (один или несколько операторов). Вариант с ключевым словом default реализуется, если ни один друго не подошёл (слово default может и отсутствовать). Если default отсутствует, а все результаты сравнения отрицательны, то ни один вариант не выполняется.

Для прекращения последующих проверок после успешного выбора некоторого варианта используется оператор break, обеспечивающий немедленный выход из переключателя switch.

Допускаются вложенные конструкции switch.

Рассмотрим правила выполнения безусловного перехода, который можно представить в следующей форме:

goto метка;

Метка - любой идентификатор, после которого поставлено двоеточие. Оператор goto указывает на то, что выполнение программы необходимо продолжать начиная с оператора, перед которым записана метка. Метку можно поставить перед любым оператором в той функции, где находится соответствующий ей оператор goto. Её не надо объявлять.

3 Функции и передача параметров; структурная декомпозиция.

Программы на языке Си обычно состоят из большого числа отделбных функций (подпрограмм). Как правило, эти функции имеют небольшие размеры и могут находится как в одном, так и в нескольких файлах. Все функции имеют небольшие размеры и могут находиться как в одном, так и в нескольких файлах. Все функции являются глобальными. В языке запрещено определять одну функцию внутри другой. Связь между функциями осуществляется через аргументы, вовзращаемые значения и внешние переменные.
В общем случае функции в языке Си необходимо объявлять. Объявление функции (т.е полное описание заголовка) должно предшествовать её использованию, а определение функции (т.е. её описание) может быть помещено как после тела программы (т.е. функции main()), так и до него. Если функция определена до тела программы, а также до её вызовов из определений других функций, то объявление может отсутствовать. Как уже отмечалось, описание заголовка функции обычно обычно называют прототипом функции.
Функция объявляется следующим образом:

тип имя_функции(тип имя_параметра_1, тип имя_параметра_2,...);

Тип функции определяет тип значения, которое вовзращает функция. Если тип не указан, то предполагается, что функция вовзращает целое значение (int).
При объявление функции для каждого её параметра можно указать только его тип (например: тип фунцкия (int, float, ...)), а может дать его имя (например: тип функция(int a, float b)). В языке Си разрешается создавать функции с переменным числом параметров. Тогда при задании прототипа вместо последнего из них указывается в многоточие.
Определение функции имеет следующий вид:

тип имя_функции(тип имя_парметра_1, тип имя_парметра_2,...)