Содержание

Вступление

CeeBot – прекрасная программа для изучения основ программирования. У нее есть свои особенности и недостатки, о которых я сейчас сообщу, но для подготовки к дальнейшему освоению сложного программирования и робототехники она подходит прекрасно.

Достоинства:

1) Возможность изучения основ языка C;
2) Возможность изучения синтаксиса языка;
3) Возможность сразу опробовать программу на виртуальном роботе;
4) Удобный редактор, который выделяет все используемые команды и подчеркивает ошибки;
5) Возможность посмотреть, как применяется любая используемая команда с примерами;
6) Возможность продолжить изучение и апробацию команд в игре Colobot Gold Edition*(о ней чуть позже)*.

Недостатки:

1) Относительно небольшое количество заданий;
2) Отсутствие свободы творчества *(если в задании линия должна быть желтая, то если он будет красная, то программа не запустится);*
3) Отсутствие полноценного обучающего курса.

Недостатки ощутимы, но все, кроме последнего решаются в продолжении Colobot Gold Edition. Поэтому данный курс будет включать в себя обе программы.

Синтаксис языка CBot

Синтаксис языка CBot, как понятно из названия, полностью взят из языка C. В данной главе мы разберем только основные моменты, глубже эта тема будет раскрыта позже.

Любая программа в CeeBot начинается со строчки «extern void object::Новая_Программа()». Но что эта строчка означает? Зачем она нужна вообще? Ответ прост: это заголовок, который рассказывает, что именно будет в данной программе. extern означает, что данная программа является основной. void, что данная программа не возвращает никаких значений *(об этом будет рассказано подробно в другой главе)*. object – тип программы (объектно-ориентированный). После :: пишется название программы. Пустые скобки говорят о том, что данная программа не принимает никаких значений *(об этом тоже будет рассказано подробно в другой главе).*

Теперь поговорим подробно о символах, используемых в CeeBot:

Символ	Значение	Пример

( )	В круглых скобках записываются операторы к командам. Так же могут использоваться для изменения порядка вычислений в примерах.	1) move(5);
		turn(90);
		2) int a=3;
		int b=4;
		(a+b)/b;

{
 	
}
	В фигурные скобки записываются действия, которые выполняются в циклах и условиях.	1) repeat(4)
		{
		move(5);
		}
		2) if (a>5)
		{
		a--;
		}
		else
		{
		a++;
		}

[ ]	Квадратные скобки нужны для создания массивов.	int a[5]; a[3]=23;
=	Операция присваивания. Заносит значения *(справа)* в переменные *(слева)*.	int a=5;

== равно
!= не равно
< меньше
<= меньше или равно
> больше
>= больше или равно
	Операции сравнения. Сравнивает то, что справа от символа с тем, что слева.	1) int a=5;
		if (a==5)
		{
		move(a);
		}
		2) int a=3;
		if (a!=4)
		{
		move(3)
		}

+, -, *, /	Стандартные математические операции плюс, минус, умножить, делить.	int a=3;
		int b=4;
		(a*a+b-a)/b;

+= -= *= /= %=	Упрощалки формул. %= *(остаток от деления)* работает только для переменных типа int.	int a=4; a+=7; // равносильно a=a+7;

!a
a && b
a || b
	Логические действия:
	Не, И, Или.
		int a=4;
		int b=8;
		if (a+b>10 && b-a<5)
		{
		  move(3);
		}

Помимо этого, важно упомянуть, что некоторые команды можно упростить. В главе «Циклы и условия» всё записывается следующим образом:

Команда цикла или условия(параметры)

{

Действия, которые повторяются или происходят при истинности условия;

}

Но эту запись можно упростить:

```
while(radar(Titanium, 0, 360, 0, 10)==null) wait(1);
```

Так как действие всего одно (*ожидание секунду*), то мы убираем фигурные скобки и пишем его в ту же стоку, что и цикл. Тем самым превращаем 4 строчки в одну. Данная в примере программа позволяет дожидаться, пока конвертер преобразует руду в титан.

Условие же можно упростить еще сильнее. Возьмем следующую запись:

```
if(a<6)
{
  move(5);
}
else
{
  move(-5);
}
```

Она занимает целых 8 строчек. Превратим её в одну строку:

```
a<6 ? move(5) : move(-5);
```

Движение

В CeeBot есть несколько команд, с помощью которых можно заставить роботов двигаться. Разберем каждую из них:

Команда	Параметры	Описание	Пример
move()	Для использования команды move() нужно указать в скобках расстояние. Можно использовать число или переменную.	Команда move() применяется для преодоления прямого участка пути. Чаще всего для небольшого отъезда бота.	1) move(10); 2) move(-3); 3) move(a);
motor()	Для использования команды motor() нужно указать в скобках через запятую скорость каждого мотора. Можно использовать числа от -1 до 1 или переменные(должны быть в тех же пределах).	Команда motor() применяется для движения по функциям. В нее можно вписать формулу движения и бот будет ей следовать. Наиболее близка к командам, применяемым в робототехнике.	1) motor(1, 1); 2) motor(-1, 1); 3) motor(a, -a); 4) motor(v+u, v-u);

goto()	Для использования команды goto() нужно указать в скобках позицию, к которой должен переместиться робот.	Команда goto() самая сложная и полезная для программ CeeBot/Colobot. Она позволяет перемещаться ботам к любым объектам. Для этого используются переменные типа point или object (в этом случае в переменную вносится объект с помощью команды radar(). После переменной ставиться .position)	1) point a(4, 7);
			goto(a);
			2) object item=radar(Titanium);
			goto(item.position);

turn()	Для использования команды turn() нужно указать в скобках угол поворота. Можно использовать число или переменную.	Команда turn() по своим свойствам аналогична move().	1) turn(90); 2) turn(-45); 3) turn(b);
jet()	Для использования команды motor() нужно указать в скобкахскорость движения робота по оси z. Можно использовать числа от -1 (вниз) до 1 (вверх) или переменные (должны быть в тех же пределах).	Команда jet() позволяет летающему боту регулировать высоту полета.	1) jet(1); 2) jet(-0,3); 3) jet(c);

Помимо этих команд, существует еще несколько, но они узкоспециализированные и не потребуются нам в дальнейшем.

Циклы и условия

В CeeBot есть три способа создания циклов и один для условий. В таблице ниже разбираем подробно каждый из них.

Команда	Параметры	Описание	Пример

repeat()
{
***
}
	Для использования цикла repeat() достаточно в круглых скобках указать требуемое количество повторений, а в фигурных то, что требуется повторять.	repeat – это самый простой способ создания цикла с ограниченным числом итераций. И самый бесполезный. В нем нельзя отслеживать, какая именно итерация сейчас выполняется.	1) repeat(4)
			{
			move(10);
			turn(90);
			}

for()
{
***
}
	Для использования цикла for в круглых скобках нужно указать следующую информацию:
	1) Объявить переменную, в которую будут записываться выполненные итерации (тип переменной int).
	2) Объявить количество итераций в формате i<5(т.е. 0, 1, 2, 3, 4).
	3) Объявить на сколько возрастает или убывает переменная при каждой итерации. Если на 1, то строчку i=i+1 можно заменить на i++.
		Цикл for позволяет делать то же, что и repeat, но при этом оставляет возможность считать итерации и использовать эти значения в своем теле. Например, если вам необходимо каждый раз ездить на 5 метров больше, то можно создать цикл из примера №2, который в первый раз проедет на 5 метров, второй на 10, третий на 15 и четвертый на 20;	1) for(inti=0; i<4; i++)
			{
			move(10);
			turn(90);
			}
			2) for(inti=0; i<4; i++)
			{
			int a=5+5*i;
			move(a);
			}

while()
{
***
}
	Для использования цикла whileв круглых скобках нужно указатьусловие, при котором цикл будет выполняться.	while– это цикл с условием. Он используется довольно часто в большинстве языков программирования. В нем так же можно считать количество итераций, но для этого нужно вводить дополнительную переменную (пример №2). Также он позволяет выполнять программу столько, сколько нужно, исходя из показаний, полученных в процессе выполнения (пример №3)	1) int a=1;
			int b=10;
			int c=22;
			while(a+b<c)
			{
			a++;
			move(2);
			}
			2) int a=20;
			int b=10;
			int c=2;
			inti=0;
			3) while(a-b!=c)
			{
			i++;
			a--;
			}
			message(i);//вывестиінаэкран
			4) while(radar(Titanium==null))

			{
			wait(1);
			}

while(true)
{
***
}
	Параметр true в цикле whileпоказывает, что цикл будет длится вечно.	while(true) нужен, когда программа должна выполняться вечно. Для остановки бесконечного цикла существует команда break.С помощью неё и команды ifон способен превратиться в цикл с условием (пример в следующей строке, №2).	1) while(true)
			{
			motor(1, 0.5);
			}
			2) while(true)
			{
			motor(1, 0.5);
			}

if()
{
***
}
	Для использования ifв круглых скобках нужно указатьусловие.	Команда ifнужна для записи условий, при которых программа выполняется. В некоторых случаях после if(если)используется еще команда else(иначе).	1) int a=3;
			int b=5;
			
			if(a>b)
			{
			message(a);
			}
			else
			{
			message(b);
			}
			2) inti=0;
			while(true)
			{
			i=i+5;
			move(i);
			if(i>50)
			{
			break;
			}
			}