CS 15900 Credit Exam

An increasing number of students entering the First Year Engineering program at Purdue University are bringing with them previous programming experience and a growing number of these students are interested in demonstrating that their experience is sufficient and transferable to earn credit for CS 15900.

What is the format of the exam?

Fifty multiple-choice questions will be selected from a large pool of problems that cover the material introduced in CS 15900. The exam will be taken on Blackboard, be limited to two hours, and the only acceptable resources for the exam are an ASCII table and an operator precedence table. Students are expected to be at a level of mathematics that includes preparation for calculus. No calculators are permitted and most of the expressions to evaluate will involve addition, subtraction, multiplication, division, and modulus.

Questions on the exam will ask students to interpret code and understand the fundamental terminology of the C programming language. There is no code writing on the exam.

What is required to pass the exam?

The credit exam is similar in content to a final exam offered in CS 15900. The average on the final exam in a typical semester is around 70% and this will be considered the threshold necessary to establish credit by exam.

What advice do you have for students who have previously programmed in another language like JAVA or C++?

The credit exam is a C language exam. The differences between programming languages at times can be minor but the details matter and this is what the exam will evaluate. A student who successfully completes CS 15900 on campus will understand these details and those attempting to establish credit by exam will be held to the same standard.

Future courses that you may take in Engineering will assume mastery of the material in CS 15900. A generic previous programming experience may be insufficient for success in future courses.

Why is my transfer credit not evaluated as equivalent to CS 15900?

The interpretation of "equivalent" has been taken to mean an identical set of topics and programming language. Most courses taken at other institutions fail to meet this mark or a level of rigor that is expected from an experience at Purdue University. Passing this credit exam will demonstrate that the experience and knowledge retained is sufficient to establish credit in CS 15900.

What other considerations should a student take into account prior to attempting the credit exam?

There is only ONE opportunity to earn the passing score. It is important to emphasize that future classes will assume an equivalent experience to CS 15900. There is no benefit to moving into a higher level course only to withdraw or earn a poor grade because the previous experience was insufficient.

A set of problems similar to those asked on the exam have been provided for your review.

- The answers are included at the end of the exam document.

What specific material is covered in CS 15900?

- From the official text of the course (ISBN-13: 978-0-534-49132-1):
  - Chapter 1 – Introduction to Computers
  - Chapter 2 – Introduction to the C Language
    - Formatting output.
  - Chapter 3 – Structure of a C Program
    - Data types – expression evaluation, type conversions.
  - Chapter 4 – Functions
    - Parameter passing paradigms.
  - Chapter 5 – Selection – Making Decisions
    - Logic issues such as short-circuit evaluation and complementary expressions.
  - Chapter 6 – Repetition
    - Iteration and recursion.
  - Chapter 7 – External File I/O
    - File functions fopen, fclose, fscanf, feof are commonly covered.
  - Chapter 8 – Arrays
    - Includes a knowledge of the searching and sorting algorithms as introduced in the text.
  - Chapter 9 – Pointers
    - Sections 9.1 and 9.2 only
  - Chapter 10 – Pointer Applications
    - Sections 10.1 – 10.4 (malloc)
  - Chapter 11 – Strings
    - Sections 11.1 – 11.5

Is there any MATLAB or Octave on the credit exam?

- **No.** The course was revised beginning with the fall semester of 2019 and is now strictly a C programming course.

CS 15900 Credit Exam Practice Questions

1. Which of the following statements regarding constants is FALSE?
   A. When evaluated in an expression both variable and constant values have data types.
   B. A literal constant is an unnamed constant used to specify data.
   C. The use of symbolic/defined constants is encouraged to give meaning to operands in an expression.
   D. None of the above.

2. Which of the following statements regarding the selection of a data type for a variable is FALSE?
   A. The operations that can be performed on a value is limited by its type.
   B. The amount of memory necessary to store a value depends on its type.
   C. How a value is stored in the memory of the computer depends on its type.
   D. None of the above.

Use the program below for problems 3 – 4

```
#include<stdio.h>

#define SHORT_PI 3.14

int main()
{
  double diameter = 2.1;
  double circumference;
  double area;
  int precision;
  int width;

  circumference = SHORT_PI * diameter;
  area = circumference * diameter;
  width = 21 - circumference;
  precision = (area - (int) area) * 5;

  printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
  printf("Circumference: %12.*lf\n", precision, circumference);
  printf("-=-=-=-=-=-=-=-=-=-=-=-=-=-\n");
  printf("Area: %*lf\n", width, area);

  return(0);
}
```

3. Which of the following is the output generated by the first two print statements in the program above?

A	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Circumference:  6.5940`	C	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Circumference:  6.5940`
B	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Circumference:  6.594`	D	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Circumference:  6.594`

4. Which of the following is the output generated by the last two print statements in the program above?

A	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Area:  13.847400`	C	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Area:  13.847400`
B	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Area:  13.8474`	D	`-=-=-=-=-=-=-=-=-=-=-=-=-=-` `Area:  13.8474`

5. Which of the following is NOT a complement of the logical expression below?

```
x == y || x != z
```

A. `!(x != y && x == z)`
B. `x != y && x == z`
C. `!(x == y || x != z)`
D. None of the above.

Use the code segment below for problems 6 – 9

```
int x = 7;
int y = 10;
int z = 5;
int result = 0;

result = ++y - 10 || z - 5 && x++;
result += y++ - 11 || z++ - 5 && x++;
result += y + 1 > 11 && (z++ >= 6 || x++);

printf("result: %d\n", result);
printf("x: %d\n", x);
printf("y: %d\n", y);
printf("z: %d\n", z);
```

6. Which of the following is the first line of output generated by the code segment above?
   A. `result: 0`
   B. `result: 1`
   C. `result: 2`
   D. None of the above.

7. Which of the following is the second line of output generated by the code segment above?
   A. `x: 7`
   B. `x: 8`
   C. `x: 9`
   D. None of the above.

8. Which of the following is the third line of output generated by the code segment above?
   A. `y: 11`
   B. `y: 12`
   C. `y: 13`
   D. None of the above.

9. Which of the following is the fourth line of output generated by the code segment above?
   A. `z: 5`
   B. `z: 6`
   C. `z: 7`
   D. None of the above.

10. Which of the following would be an invalid user-defined function name?
    A. `_123`
    B. `printf_`
    C. `1stcall`
    D. None of the above

11. Which of the following statements regarding user-defined functions is FALSE?
    A. A variable declared in the local declaration section of a function can have the same identifier as one of the parameters within the same function.
    B. Data sent from the calling function to the function being called will be received in the same order in which it was passed.
    C. Parameters are defined as local variables in the first line of the function definition and should not be re-defined within the local declaration section of the function.
    D. None of the above.

12. Which of the following statements regarding the short-circuit method of evaluating logical expressions is TRUE?
    A. The truth value of a logical expression will not change whether a short-circuit method occurs or not.
    B. The short-circuit method is limited to use in selection constructs.
    C. The short-circuit method is limited to use in repetition constructs.
    D. None of the above.

13. Which of the following statements regarding `if` constructs is FALSE?
    A. Proper indentation of `else` will determine to which `if` it belongs in a nested construct.
    B. Nested `if` constructs can be used to test different variables.
    C. While there is no limit to the number of levels of nesting in a nested `if` construct, a larger number of levels may make the code difficult to read.
    D. None of the above.

14. Which of the following statements regarding the conditional expression is TRUE?
    A. Only a single terminal semicolon is used to terminate a conditional expression.
    B. A conditional expression cannot have another conditional expression as one of its executable actions.
    C. The logical expression of a conditional expression is limited to operands of the integer and character data types.
    D. None of the above.

15. Which of the following statements regarding the rules of the `switch` construct is TRUE?
    A. Two `case` labels can have the same constant expression value.
    B. No two `case` labels can be associated with the same set of actions.
    C. The control expression that follows the keyword `switch` must be an integral type.
    D. None of the above.

16. Which of the following statements regarding file functions is TRUE?
    A. The `fscanf` function requires the name of the external data file as one of its parameters.
    B. The `fopen` function requires the name of the external data file as its only parameter.
    C. The `fclose` function requires the name of the external data file as its only parameter.
    D. None of the above.

17. When passing a multi-dimensional array to a function where would you NOT include the sizes of those dimensions beyond the first?
    A. In the declaration of the function being called.
    B. In the definition of the function being called.
    C. In the call of the function.
    D. None of the above.

18. Which of the following statements regarding arrays and user-defined functions is FALSE?
    A. It is possible to pass multiple individual elements of an array to a function by value.
    B. It is possible to pass individual elements of an array to a function by address.
    C. It is possible to pass a whole array to a function by value.
    D. None of the above.

19. Which of the following statements regarding arrays is FALSE?
    A. The memory address represented by the name of an array can change during the execution of a program.
    B. When adding an integer to the name of an array the result is a value that corresponds to another index location.
    C. The dereference of an array name is the value of its first element.
    D. None of the above.

Use the program below for problems 20 – 21

```
#include<stdio.h>

int numDaysInMonth(int);

int main()
{
  int i;
  int num = 0;

  for(i = 1; i < 6; i++)
  {
    num += numDaysInMonth(i);
  }

  printf("num: %d\n", num);

  return(0);
}

int numDaysInMonth(int month)
{
  int days = 0;

  switch(month % 2)
  {
    case 1:  days = 31;
             break;
    case 0:  days = month == 2 ? 28 : 30;
  }

  return(days);
}
```

20. Which of the following is the output generated by the print statement in the program above?
    A. `num: 151`
    B. `num: 152`
    C. `num: 154`
    D. None of the above.

21. Which of the following is would be the output generated by the print statement in the program above if the control expression of the switch were changed from `(month % 2)` to `(month % 2 == 0)` ?
    A. `num: 151`
    B. `num: 152`
    C. `num: 154`
    D. None of the above.

Use the code segment below for problems 22 – 23

```
int div;
int x = 30;
int total = 0;

while(x <= 41)
{
  div = 2;

  while(x % div != 0 && div <= sqrt(x))
  {
    div++;
  }

  if(div > sqrt(x))
  {
    total += div;
  }

  x++;
}

printf("total: %d\n", total);
printf("div: %d\n", div);
```

22. Which of the following is the output generated by the first print statement in the code segment above?
    A. `total: 20`
    B. `total: 21`
    C. `total: 25`
    D. None of the above.

23. Which of the following is the output generated by the second print statement in the code segment above?
    A. `div: 2`
    B. `div: 6`
    C. `div: 7`
    D. None of the above.

Use the code segment below for problems 24 – 25

```
int a;
int b;
int c;
int d = 0;

for(a = 64; a > 1; a /= 2)
{
  for(b = 1; b < 27; b *= 3)
  {
    for(c = 2; c < 10; c += 2)
    {
      d++;
    }
  }
}

printf("sum: %d\n", a + b + c);
printf("d: %d\n", d);
```

24. Which of the following is the output generated by the first print statement in the code segment above?
    A. `sum: 40`
    B. `sum: 38`
    C. `sum: 36`
    D. None of the above.

25. Which of the following is the output generated by the second print statement in the code segment above?
    A. `d: 84`
    B. `d: 66`
    C. `d: 60`
    D. None of the above.

Use the code segment below for problem 26

```
int x[5] = {6, 9, 3, 0, 4};
int y[5] = {0};

y = x;

printf("y[0] = %d\n", y[0]);
```

26. Which of the following describes the integer value generated by the print statement in the code segment above?
    A. The value displayed will be the integer 6.
    B. The value displayed will be the memory address represented by the array `x`.
    C. No integer value will be displayed due to a compiler error regarding the assignment statement.
    D. None of the above.

Use the code segment below for problems 27 – 28

```
int findValues(int);

int main()
{
  int val;

  val = findValues(154);

  printf("val: %d\n", val);

  return(0);
}

int findValues(int n)
{
  int seek = 0;

  if(n > 0)
  {
    seek = findValues(n / 5);

    if(n % 2 == 0 && n % 10 > seek)
    {
      seek = n % 10;
    }
  }
  return(seek);
}
```

27. Which of the following is the output generated by the print statement in the code segment above?
    A. `val: 4`
    B. `val: 1`
    C. `val: 6`
    D. None of the above.

28. What is the total number of times that the `findValues` function is called in the code segment above?
    A. Four
    B. Five
    C. Six
    D. None of the above.

29. Which of the following statements regarding pointer variables is FALSE?
    A. Working with an uninitialized pointer variable is a compile-time error.
    B. The asterisk character (`*`) is used in two different contexts for pointer variables; for declaration and for dereferencing.
    C. The value of a pointer variable can change during the execution of a program.
    D. None of the above.

30. Which of the following statements regarding dynamic memory allocation is TRUE?
    A. The result of the `malloc` function is assigned to an array variable.
    B. The value passed to the `malloc` function is the product of the number of elements to store and the amount of memory required to store a value of the given data type.
    C. The memory allocated as a result of the `malloc` function is initialized to a default value based on the data type specified.
    D. None of the above.

Use the program below for problems 31 – 33

```
#include<stdio.h>

#define SIZE 8

int searchArray(int[], int);

int main()
{
  int x[SIZE] = {9, 5, 15, 3, 14, 17, 7};
  int y[SIZE] = {11, 15, 14, 17, 16, 0, 5};
  int i;
  int ct = 0;

  for(i = 0; i < SIZE; i++)
  {
    ct += searchArray(y, x[i]);
  }

  printf("ct: %d\n", ct);
  printf("y[0]: %d\n", y[0]);
  printf("y[5]: %d\n", y[5]);

  return(0);
}

int searchArray(int arry[], int n)
{
  int i;
  int result = 0;

  for(i = 0; i < SIZE; i++)
  {
    if(arry[i] == n)
    {
      result = 1;
      arry[i] = -1;
      i = SIZE;
    }
  }

  return(result);
}
```

31. Which of the following is the output generated by the first print statement in the program above?
    A. `ct: 0`
    B. `ct: 4`
    C. `ct: 5`
    D. None of the above.

32. Which of the following is the output generated by the second print statement in the program above?
    A. `y[0]: 0`
    B. `y[0]: -1`
    C. `y[0]: 11`
    D. None of the above.

33. Which of the following is the output generated by the third print statement in the program above?
    A. `y[5]: 0`
    B. `y[5]: -1`
    C. `y[5]: 16`
    D. None of the above.

Use the program below for problems 34 – 36

```
#include<stdio.h>

void changeArray(int[], int);

int main()
{
  int x[8] = {2, 3, 5, 4, 1, 0, 7, 6};
  int i;

  for(i = 0; i < 8; i += 2)
  {
    changeArray(x, i);
  }

  printf("x[0] = %d\n", x[0]);
  printf("x[3] = %d\n", x[3]);
  printf("x[7] = %d\n", x[7]);

  return(0);
}

void changeArray(int y[], int i)
{
  int j;

  for(j = 1; j < 8 - i; j++)
  {
    y[j] += y[j - 1];
  }
}
```

34. Which of the following is the output generated by the first print statement in the program above?
    A. `x[0] = 8`
    B. `x[0] = 5`
    C. `x[0] = 2`
    D. None of the above.

35. Which of the following is the output generated by the second print statement in the program above?
    A. `x[3] = 26`
    B. `x[3] = 31`
    C. `x[3] = 57`
    D. None of the above.

36. Which of the following is the output generated by the third print statement in the program above?
    A. `x[7] = 28`
    B. `x[7] = 29`
    C. `x[7] = 30`
    D. None of the above.

Use the program below for problems 37 – 39

```
#include<stdio.h>

#define SIZE 9

int main()
{
  int i;
  int x[SIZE] = {13, 11, 22, 32, 15, 23, 28, 19, 18};

  for(i = 0; i < SIZE; i++)
  {
    x[i] -= x[SIZE - i - 1];
    x[SIZE - i - 1] += x[i];
    x[i] = x[SIZE - i - 1] - x[i];
  }

  printf("x[2] = %d\n", x[2]);
  printf("x[4] = %d\n", x[4]);
  printf("x[8] = %d\n", x[8]);

  return(0);
}
```

37. Which of the following is the output generated by the first print statement in the program above?
    A. `x[2] = 22`
    B. `x[2] = 28`
    C. `x[2] = 44`
    D. None of the above.

38. Which of the following is the output generated by the second print statement in the program above?
    A. `x[4] = 0`
    B. `x[4] = 15`
    C. `x[4] = 30`
    D. None of the above.

39. Which of the following is the output generated by the third print statement in the program above?
    A. `x[8] = 13`
    B. `x[8] = 18`
    C. `x[8] = 26`
    D. None of the above.

Use the code segment below for problem 40

```
int x = 3;
int *y;
int *z;

y = &x;
z = y;
(*z)--;

printf("result: %d\n", *y + *z);
```

40. Which of the following is the output generated by the print statement in the code segment above?
    A. `result: 6`
    B. `result: 5`
    C. `result: 4`
    D. None of the above.

Use the program below for problems 41 – 43

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

int main()
{
  char str1[21] = "ABC Company";
  char str2[31] = "ABC Corp";
  int result;

  result = strcmp(str1, str2);
  printf("result #1: %d\n", result);

  result = strlen(str2);
  printf("result #2: %d %d\n", result, str2[result]);

  strcpy(str1, str2);
  printf("str1: %s str2: %s\n", str1, str2);

  return(0);
}
```

41. Which of the following is the output generated by the first print statement in the program above?
    A. `result #1: 5`
    B. `result #1: 0`
    C. `result #1: -5`
    D. None of the above.

42. Which of the following is the output generated by the second print statement in the program above?
    A. `result #2: 8 0`
    B. `result #2: 8 112`
    C. `result #2: 7 112`
    D. None of the above.

43. Which of the following is the output generated by the third print statement in the program above?
    A. `str1: ABC Corp str2: ABC Corp`
    B. `str1: ABC Company str2: ABC Company`
    C. `str1: ABC Corpany str2: ABC Corp`
    D. None of the above.

44. Given the array below:

    14 21 13 12 20 15

    and the array after two passes through a sorting algorithm:

    12 13 14 21 15 20

    identify which of the following algorithms has been applied.

    A. Selection Sort
    B. Bubble Sort
    C. Insertion Sort
    D. More than one of the above.

45. Which of the following would be the final values of the `first` and `last` variables when using the binary search with the array below given a target value of 45?

```
13   16   20   24   27   30   31   35   36   41
```

A. `first = 10 last = 9`
B. `first = 9 last = 8`
C. `first = 9 last = 9`
D. None of the above.

46. Given the array below:

```
14   21   13   12   20   15
```

Which of the following is NOT a possible configuration of the array after two passes through the selection sorting algorithm?

A. `{21, 20, 13, 12, 14, 15}`
B. `{14, 21, 20, 15, 13, 12}`
C. `{14, 12, 13, 15, 20, 21}`
D. None of the above.

Use the program below for problems 47 – 48

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

#define SIZE 50

int main()
{
  char str[SIZE] = "TEAETEAEAEAT";
  char *ptr;
  int ct = 0;

  ptr = strstr(str, "AEA");

  printf("Result: %s\n", ptr);

  ptr = strstr(str, "EAE");

  while(ptr != NULL)
  {
    ct++;
    ptr = strstr(ptr + 1, "EAE");
  }

  printf("Final ct value: %d\n", ct);

  return(0);
}
```

47. Which of the following is the output generated by the first print statement in the program above?
A. `Result: EAEAT`
B. `Result: AEA`
C. `Result: AEAEAT`
D. None of the above.

48. Which of the following is the output generated by the second print statement in the program above?
A. `Final ct value: 7`
B. `Final ct value: 2`
C. `Final ct value: 3`
D. None of the above.

49. Which of the following statements regarding type conversions is FALSE?
    A. When the types of two operands for the division operator are different, the lower-ranked type is promoted to the rank of the higher type before the quotient is determined.
    B. An explicit type conversion is the programmer taking control and determining the data type of an operand in an expression.
    C. In an assignment statement, promotion occurs if the right expression has a lower rank than the variable on the left and demotion occurs if the right expression has a higher rank.
    D. None of the above.

50. Which of the following statements regarding index range checking in arrays is FALSE?
    A. The issue of attempting to access an index beyond the range of valid index values for an array is a logical error.
    B. It is possible that when attempting to access an index beyond the range of valid index values for an array that the output results may not be as expected.
    C. The problem of index range checking is limited to only those values that are greater than the largest index values for an array.
    D. None of the above.

51. Which of the following statements regarding the delimiter character and character arrays is FALSE?
    A. The delimiter character is used to separate the data in the array from the unused elements of the array.
    B. The delimiter character is not needed when the amount of data present is equal to the capacity of the array.
    C. The capacity of the array must account for the data it is to store and an extra space for the delimiter character.
    D. None of the above.

52. Which of the following statements regarding the sorting algorithms introduced this semester is FALSE?
    A. After completing the first pass of the insertion sorting algorithm there are two values in the sorted list.
    B. On the final pass of the bubble sorting algorithm there are two values brought into the sorted listed.
    C. A value placed into the sorted list during the process of the selection sort will not move again on any remaining passes.
    D. None of the above.

53. Which of the following statements regarding the binary searching algorithm is TRUE?
    A. The binary searching algorithm will always find a target in an array faster than the sequential searching algorithm.
    B. The binary searching algorithm should always be used for searching.
    C. With each comparison made in the binary search approximately half of the remaining elements in the array are eliminated as possible locations of the target.
    D. None of the above.

54. Which of the following statements regarding the `scanf` statement is TRUE?
    A. A single `scanf` statement cannot be used to accept input for multiple values.
    B. Each use of the `scanf` function will contain both a format string and address list.
    C. It is possible to make use of the `scanf` function to both accept input and generate output.
    D. None of the above.

ASCII Table

Char	Dec	Char	Dec	Char	Dec	Char	Dec
delimiter	0	space	32	@	64	`	96
(soh)	1	!	33	A	65	a	97
(stx)	2	"	34	B	66	b	98
(etx)	3	#	35	C	67	c	99
(eot)	4	$	36	D	68	d	100
(enq)	5	%	37	E	69	e	101
(ack)	6	&	38	F	70	f	102
(bel)	7	'	39	G	71	g	103
(bs)	8	(	40	H	72	h	104
(ht)	9	)	41	I	73	i	105
(nl)	10	*	42	J	74	j	106
(vt)	11	+	43	K	75	k	107
(np)	12	,	44	L	76	l	108
(cr)	13	-	45	M	77	m	109
(so)	14	.	46	N	78	n	110
(si)	15	/	47	O	79	o	111
(dle)	16	0	48	P	80	p	112
(dc1)	17	1	49	Q	81	q	113
(dc2)	18	2	50	R	82	r	114
(dc3)	19	3	51	S	83	s	115
(dc4)	20	4	52	T	84	t	116
(nak)	21	5	53	U	85	u	117
(syn)	22	6	54	V	86	v	118
(etb)	23	7	55	W	87	w	119
(can)	24	8	56	X	88	x	120
(em)	25	9	57	Y	89	y	121
(sub)	26	:	58	Z	90	z	122
(esc)	27	;	59	[	91	{	123
(fs)	28	<	60	\	92	|	124
(gs)	29	=	61	]	93	}	125
(rs)	30	>	62	^	94	~	126
(us)	31	?	63	_	95	(del)	127

This page lists C operators in order of *precedence* (highest to lowest). Their *associativity* indicates in what order operators of equal precedence in an expression are applied.

Operator	Description	Associativity

`()`
`[]`
`++ --`
	Parentheses (function call)
	Brackets (array subscript)
	Postfix increment/decrement
		left-to-right

`++ --`
`+ -`
`!`
`(`*`type`*`)`
`*`
`&`
`sizeof`
	Prefix increment/decrement
	Unary plus/minus
	Logical negation
	Cast (change *type*)
	Dereference
	Address
	Determine size in bytes
		right-to-left

`* / %`	Multiplication/division/modulus	left-to-right
`+ -`	Addition/subtraction	left-to-right
`< <=` `> >=`	Relational less than/less than or equal to Relational greater than/greater than or equal to	left-to-right
`== !=`	Relational is equal to/is not equal to	left-to-right
`&&`	Logical AND	left-to-right
`||`	Logical OR	left-to-right
`? :`	Conditional expression	right-to-left

`=`
`+= -=`
`*= /=`
`%=`
	Assignment
	Addition/subtraction assignment
	Multiplication/division assignment
	Modulus assignment
		right-to-left

`,`	Comma (separate expressions)	left-to-right