1 True or False 1 point

In C, you can access out-of-bound elements of an array without any error being reported by the compiler.

- (●) True
- ( ) False

2 Multiple Answer 1 point

Which command initializes the following 2 x 2 identity matrix:

```
1 0
0 1
```

- [x] `double foo[2][2] = {[0][0] = 1.0, [1][1] = 1.0};`
- [x] `double foo[2][2] = {1.0, 0., 0., 1.0};`
- [ ] `double foo[2][2] = {1.0, 0., 1.0, 0.0};`
- [ ] `double foo[2][2] = {'10','01'};`

3 Multiple Choice 1 point

What must happen before a function can be called?

- (●) It must be declared or defined
- ( ) It will automatically be recognized by the compiler
- ( ) It does not affect the compilation
- ( ) It causes the program to exit

4 Multiple Choice 1 point

What is the purpose of C functions?

- (●) To reuse, recall, and modularize code
- ( ) To create graphical user interfaces
- ( ) To manage database connections
- ( ) To enhance the speed of the processor

5 Multiple Choice 1 point

What does the `void` type indicate in a function definition?

- (●) The function does not return a value
- ( ) The function returns an integer
- ( ) The function can return any type of value
- ( ) The function must take at least one argument

6 True or False 1 point

The `main` function can be omitted in any C program.

- ○ True
- ● False

7 Matching 1 point

Match the function declaration to the return type and the required arguments:

void foo()	no return value, takes no arg
int foo(float x)	returns integer, takes single
double foo(char c)	returns double, takes charac
char foo(int a)	returns character, takes inte

Possible answers

- returns integer, takes single floating-point argument
- no return value, takes no arguments
- returns character, takes integer argument
- returns double, takes character argument

8 Multiple Choice 1 point

What does * do when applied to an address like `&i` (where `i` is a variable)?

- (●) It dereferences the address to access the value.
- ( ) It converts the address to a hexadecimal format.
- ( ) It increases the memory address by one step.
- ( ) It checks the validity of the address.

9 Multiple Choice 1 point

What does &*p in C programming return if p is a pointer?

- (●) The address of the variable that p points to.
- ( ) The value pointed by p.
- ( ) A new pointer that points to p.
- ( ) An error unless p is a pointer array.

10 Multiple Choice 1 point

If p is a pointer to an integer, what does `*p = 10;` do?

- (●) Assigns the value 10 to the location p points to.
- ( ) Makes p point to the memory address 10.
- ( ) Increases the value at p by 10.
- ( ) Causes a compilation error unless p is initialized.

11 True or False 1 point

You can change the address that a pointer variable holds after it is initialized.

- (●) True
- ( ) False

12 Matching 1 point

Matching Question Match the pointer operation with its correct description:

*p	Dereferences a pointer to a
&var	Gets the address of a variab
p = &var;	Assigns a new address to a p
*&i	Gets the value of i by derefe

Possible answers

- Assigns a new address to a pointer.
- Gets the value of i by dereferencing an address
- Dereferences a pointer to access the value it points to.
- Gets the address of a variable.

13 Multiple Choice 1 point

How is an array typically passed to a function in C?

- (●) By passing the name of the array, which decays to a pointer to its first element.
- ( ) By passing the size of the array only.
- ( ) By passing each element of the array individually.
- ( ) By copying the array into a new pointer variable.

14 Multiple Answer 1 point

Which of these methods can be used to initialize an array `a`?

(More than one answer is correct).

- [x] Initialize individual elements, like `a[3] = 1;`
- [x] Initialize using a `for` loop and `scanf` to read from input
- [x] Initialize and declare as list, as in int `a[ ] = {1,2,3,4,5};`
- [ ] Arrays in C are automatically initialized.

15 Multiple Answer 1 point

Which statement computes the length of the array, declared as: `int a[4]`?
(More than one answer is correct.)

- [x] `sizeof(a) / sizeof(a[0])`
- [ ] `sizeof(a)`
- [ ] `sizeof(a) * sizeof(a[0])`
- [x] `sizeof(a) / sizeof(a[3])`

16 True or False 1 point

The two loops in this program have different outputs:

```
#include <stdio.h>
int main() {

  for (int i = 0; i++ < 5; ) printf("%d ",i);
  puts("");
  for (int i = 0; ++i < 5; ) printf("%d ",i);

  return 0;
}
```

- (●) True
- ( ) False

17 Multiple Choice 1 point

What is the error in this program?

The program does not compile. What's wrong?

```
#include <stdio.h>
int main() {
    for (int i = 1; i < 10; i++);
    i++;
    return 0;
}
```

- (●) `i` is undeclared (first use in this function in the `i++`statement)
- ( ) You cannot declare `i` inside the for loop
- ( ) There should not be a semi-colon ; after the `for` statement
- ( ) Brackets `{...}` are missing around the `i++` statement

18 Matching 1 point

Match the pattern and the code

Counting up from 0 to n-1	for (i = 0; i < n; i++)
Counting up from 1 to n	for (i = 1; i <= n; i++)
Counting down from n-1 to 0	for (i = n-1; i >= 0; i--)
Counting down from n to 1	for (i = n; i > 0; i--)

Possible answers

- for (i = 1; i <= n; i++)
- for (i = n; i > 0; i--)
- for (i = 0; i < n; i++)
- for (i = n-1; i >= 0; i--)

19 Multiple Answer 1 point

Fix the program to get the output

This program is supposed to print out 12345 but it does not print out anything and instead runs forever. Pick the corrected expressions that would fix the program below.

```
int i = 1;
while (i <= 5) {
  printf("%d", i);
}
```

More than one answer is correct.

- [x] `printf("%d", i++);`
- [x] `printf("%d", i); i++;`
- [ ] `int i += 1;`
- [ ] `while (i++ <= 5)`

20 Multiple Choice 1 point

What output does this program produce?

Tip: remember that `i *= 2` stands for `i = i * 2`.

```
int i = 1;
while ( i <= 8 ) {
  printf("%d ", i);
  i *= 2;
 }
```

- (●) `1 2 4 8`
- ( ) `2 4 8 16`
- ( ) `2 2 2 2`
- ( ) No output

21 Multiple Choice 1 point

Given the following C code snippet, what will be the output if `x = 15`?

```
int x = 15;
if (x > 10)
    printf("Greater than 10\n");
else if (x < 20)
    printf("Less than 20\n");
else
    printf("Neither\n");
```

- (●) Greater than 10
- ( ) Less than 20
- ( ) Neither
- ( ) No output

22 True or False 1 point

The statement `while(1)` leads to an infinite loop

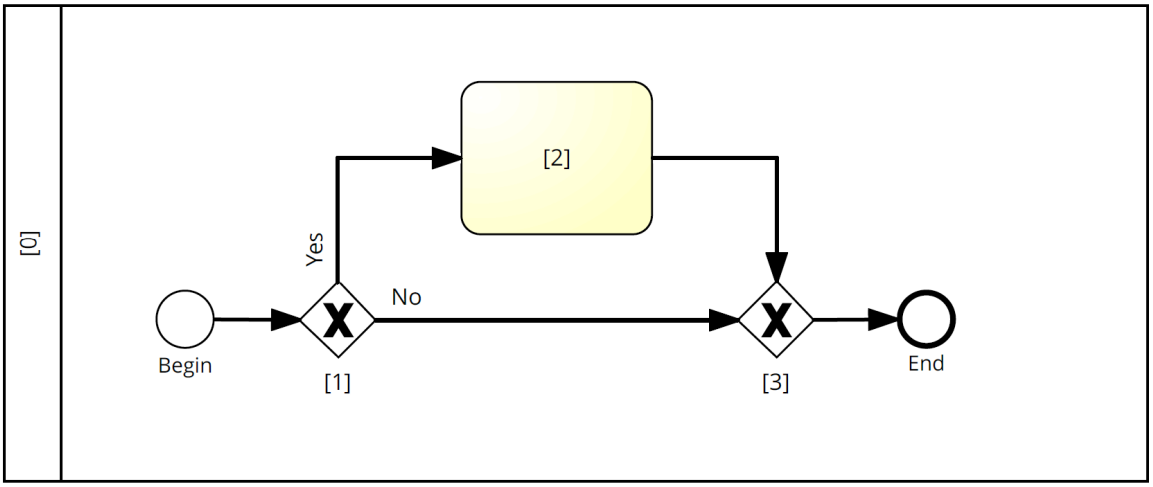
- (●) True
- ( ) False

23 Multiple Choice 1 point

Complete the BPMN model shown below to match the pseudocode

Replace [0], [1], [2], and [3] in the BPMN model below so that the model describes this pseudocode:

```
if Date == April 18
   File taxes
```



- [x] [0] Tax payer
  [1] Is the date April 18?
  [2] File taxes
  [3]
- [ ] [0] IRS
  [1] Date April 18
  [2] File taxes
  [3] Done
- [ ] [0] Tax payer
  [1] Date != April 18?
  [2] Defer taxes
  [3] Date != April 18
- [ ] [0] Birkenkrahe
  [1] Date == April 18?
  [2] File taxes
  [3]

24 True or False 1 point

Pseudocode needs to compile and run.

- ○ True
- ● False

25 Multiple Choice 1 point

In C, how is a Boolean condition represented?

- ● True as 1 and false as 0
- ○ With specific TRUE and FALSE types
- ○ Using the words `True` and `False` directly
- ○ By using the symbols 1 and 0 in parentheses

26 Matching 1 point

Match Boolean operator and the corresponding logical operator in C.

NOT	!
AND	&&
OR	||

Possible answers

! || &&

27 Multiple Choice 1 point

What must the controlling expression of a switch statement in C be?

- (●) An integer expression
- ( ) A string expression
- ( ) A floating-point number
- ( ) A Boolean expression

28 Multiple Choice 1 point

When making models via abstraction, it is advised to:

- (●) Stay as close to the problem description as possible
- ( ) Deviate significantly from the problem description
- ( ) Ignore the logic of the problem
- ( ) Always use complex logic to impress the client

29 Multiple Choice 1 point

Which of the following is NOT an operator in C? Here, a and b are variables.

- (●) `a//b`
- ( ) `a%b`
- ( ) `a==b`
- ( ) `++a`
- ( ) `-b`

30 Multiple Choice 1 point

If `int i = 100` and `int j = 50`, what will the expression `!i < j` print in C?

*Tip: remember that, in C, the **logical** value of any number but 0 is true displayed as 1, or `!0 == 1`.*

- (●) 1
- ( ) 0
- ( ) FALSE
- ( ) TRUE

31 Matching 1 point

Match the expression and the C operator type

Arithmetic	i * j + k
Relational	i > j
Logical	i && j
Assignment	i = j

Possible answers

i && j | i = j | i * j + k | i > j

32 True or False 1 point

The Statement `if (i = 0)` always fails.

- (●) True
- ( ) False

33 True or False 1 point

C evaluates all non-zero values as `TRUE`, and all zero values as `FALSE`

- (●) True
- ( ) False

34 Multiple Choice 1 point

What's wrong with this code?

The code below returns false output because of the conditional expression `(i < j < k)`. What's the fix?

```
int i = 5, j = 1, k = 100;
if ( i < j < k ) {
  puts("TRUE");
 } else {
  puts("FALSE");
 }
```

```
TRUE
```

- (●) `( i < j && j < k )`
- ( ) `( i < j || j < k )`
- ( ) `( k < i < j )`
- ( ) `( i > j && j > k )`

35 Matching 1 point

Match the compound operator expression to the expected result.

Tip: `a -= b` is the same as the assignment `a = a - b`

10 += 2	12
10 *= 2	20
10 /= 2	5
10 %= 2	0

Possible answers

20 12 0 5

36 Multiple Choice 1 point

This code asks for a floating-point input `x`, for example `1.67`. It compiles, and it accepts the input but it prints nothing. What is wrong?

```
float x;
scanf("%f", x);
printf("Your input was: %.2f", x,"\n");
```

- (●) The `scanf` function requires `&x` as argument
- ( ) The `printf` function is wrongly formatted
- ( ) You cannot have `printf` and `scanf` in the same program
- ( ) The `scanf` function only accepts integers

37 Multiple Choice 1 point

The code below is supposed to print this:

```
: Speed of light (m/s): c = 299792458
: Euler number: e = 2.718282
```

But instead it prints this:

```
: Speed of light (m/s): c = 0.000000
: Euler number: e = -1610612736
```

Fix the program to get the correct output!

```
int c = 299792458;
float e = 2.718282f;

printf("Speed of light (m/s): c = %f\n", c);
printf("Euler number: e = %d\n", e);
```

- (●) Swap the format specifiers in the print statements
- ( ) Declare c as floating point number, and e as integer
- ( ) Remove `%f` and `%d` from the print statements
- ( ) Define c and e as constants

38 Matching 1 point

Help me escape!

Match the `printf()` statement and the output.

printf("hi there");	hi there
printf(" \"hi there\" ");	"hi there"
printf("This is a slash: \ ");	This is a slash:
printf("This is a slash: \\ ");	This is a slash: \

Possible answers

This is a slash: | This is a slash: \ | "hi there" | hi there

39 Multiple Answer 1 point

Which of these are not legal identifiers?

- [x] 100_bottles
- [x] bottles-100
- [ ] _100_bottles
- [ ] one__hundred__bottles

40 Multiple Answer 1 point

Be the compiler! Identify the choices that would fix the program.

The program should print

```
foo is 100, and bar is 200
```

but it won't even compile! What's wrong?

*Tip: there are **two** things wrong with this program.*

```
#include <stdio>

float main(void) {

  int foo = 100, bar;
  printf("foo is %d, and bar is %d\n", foo, bar = 200);

  return 0;
}
```

- [x] The pre-processor declaration should include the header file 'stdio.h'
- [x] The main() function should be 'int' and not 'float'
- [ ] You cannot assign a value to 'bar' inside the printf() function
- [ ] You cannot assign a value to 'foo' when declaring the type as 'int'

41 Multiple Choice 1 point

Reading input

Suppose that we call `scanf` as follows:

```
scanf("%f%d%f", &x, &i, &y);
```

If the user enters

```
12.3 45.6 789
```

What will be the values of `x`, `i`, and `y` after the call (if we print with the correct format specifiers)? Assume that `x` and `y` are uninitialized `float` variables, and `i` is an uninitialized `int` variable.

- (●) `x=12.300000, i=45, y=.600000`
- ( ) `x=12.300000, i=45, y=789.000000`
- ( ) `x=12.3, i=45.789, y=0.000000`
- ( ) `x=12.30, i=45, y=789.00`

42 Multiple Choice 1 point

Which format specifier prints the number -12345 on 8 places, aligning the number on the right of the reserved places?

- (●) `%8d`
- ( ) `%-8d`
- ( ) `%-7.2d`
- ( ) `%9d`

43 Multiple Answer 1 point

What is Emacs?

- [x] A self-extensible text editor
- [x] Free and open source software
- [x] A literate programming environment
- [ ] A Microsoft Office program

44 Matching 1 point

Both images below show the same section of a file (an Emacs configuration file, `.emacs`). In one file, the **syntax** is **highlighted** to aid understanding and coding, in the other it is not.

Which image exhibits syntax highlighting, and which does not?

Image 1:

Image 2:

1	Image does show syntax hig
2	Image does not show syntax

Possible answers

Image does show syntax highlighting

Image does not show syntax highlighting

45 Multiple Answer 1 point

What are some of the reasons for C's long-lasting impact and success?

- [x] Its expressiveness, efficiency, and the environment it was created in (Unix)
- [x] Its power and the ability to manage memory directly
- [ ] Its simplicity and readability
- [ ] Its extensive use in mobile and web application development

46 Multiple Answer 1 point

Describe the main components of a computer's hardware relevant to programming in C.

- [x] The CPU, which processes instructions
- [x] RAM, which stores data temporarily
- [x] Non-volatile memory (like a hard disk), which provides permanent storage
- [ ] Sound card, which produces audio output

47 Categorization 1 point

Assign C's strengths and weaknesses correctly to the two categories. Note: Not all properties are strengths or weaknesses of C!

Strengths

- Efficiency and speed
- Portability
- Integration with UNIX

Weaknesses

- Permissiveness (error-prone)
- Terseness and understanding
- Large program maintenance

Possible answers

- Algorithms
- Emacs
- Compiler

48 True or False 1 point

C contains the object-oriented programming (OOP) language C++

- ○ True
- ● False

49 Multiple Choice 1 point

What is gcc?

- ○ A graphical user interface
- ○ A file format
- ◉ A program to compile source code to machine code
- ○ A computer operating system

50 Multiple Choice 1 point

What is the main characteristic of "literate programming"?

- ○ Programming with a focus on technical specifications
- ○ Programming that is only machine-readable
- ◉ Programming that includes human-readable documentation and storytelling
- ○ Programming in literary style