C quiz sample questions

Q1

What does IDE stand for?

a) Integrated Development Environment

b) Intermediate Developer Environment

c) Integrated Design Executable

d) Intelligent Design Elements

Q2

Which function from the time standard C library can be used to obtain the current time?

a) getTime()

b) time()

c) timenow()

d) null(TIME)

Q3

What type of variable will the following function output?

printf("23 + 1");

a) String Array

b) Float

c) Int

d) Char Array

Q4

Type the exact output from the following code:

```
int a = 0;
if(a <= 0)
{
    printf("Win");
}
if(a >= 0)
{
    a++;
    printf("Lose");
}
printf("%d", a);
```

Q5

Type the exact output of the following code:

```
int GetX()
{
    int x = 11;
    return x;
}

int main()
{
    int x = 10;
    x += GetX();
    printf("%d", x);
    return(0);
}
```

Answers

Q1

a) Integrated Development Environment

Q2

b) time()

Explanation:

Watch video 4.9: Working with time functions

Or look online for time.h definitions and examples. This header file is quite useful.

https://fresh2refresh.com/c-programming/c-time-related-functions/

Q3

d) Char Array

Explanation:

Quotation marks are a good indicator that the printed value is either a string or a single character e.g. printf("23 + 1");

If the answer was a b) integer or c) float, then the code would look like this: printf("%d",23+1) or printf("%i",23+1).

This leaves a) and d). From a high-level coding point of view both are correct. However, the question is asking for a variable type. C does not have a specific string variable type. Therefore, it must be a character array.

Q4

`WinLose1`

Explanation:

Let's go through the code.

a=0.

On the first 'if' statement, we're checking if a is less than or equal to 0. This condition is met. Therefore, the code inside is compiled. The code is `printf("Win")`. This will print "Win" onto the command window. This would now look like

`Win`

On the second 'if' statement, we're checking if a is greater than or equal to 0. a is still equal to zero. Therefore, the code inside is also compiled. This code is a little bit different. Firstly, there is an "`a++`" line. This operates as basically increase the value of a by 1 (a=a+1). After this line a=0+1=1. Then the `printf("Lose")` line is compiled. This will print "Lose" onto the command window. This would now look like:

`WinLose`

But Howe, why doesn't it look like these?

```
Win Lose
```

Or

```
Win

Lose
```

Good question, this is because the C compilers doesn't make any assumptions. If you don't put a space between the strings then the compiler won't add a space. If you don't put a carriage return (\n) in the strings then the compiler won't add a carriage return.

Now the final line is:

```
printf("%d", a);
```

This will print out the value of a, which is a=1.

Which results in:

```
WinLose1
```

Q5

```
21
```

Explanation:

Oh no this code has a function it. Don't panic, functions are a part of life. Functions only work when they are called. When we are looking through each line, don't worry about the function until you see the name pop up in the main function. Now let's look at the main function.

It starts with x=10. Then the line "`x+= GetX();`" pops up. The operator "+=" functions as an addition operator (basically x=x+GetX()). Now you will need to look at GetX() function to figure out how much to add to x.

GetX() starts with x=11 and then returns x. This means the GetX() will be recognised as the value of x, which in this case is 11.

Subbing that back into the main function will result in x=x+11=10+11=21.

The next line will then print out x, which results in the value 21 being printed.