

C++ Institute

CLA-11-03

CLA - C Certified Associate Programmer

For More Information – Visit link below:

<https://www.examsempire.com/>

Product Version

1. Up to Date products, reliable and verified.
2. Questions and Answers in PDF Format.



<https://examsempire.com/>

Visit us at: <https://www.examsempire.com/cla-11-03>

Latest Version: 6.0

Question: 1

Assume that ints and floats are 32-bit wide.

What happens if you try to compile and run this program?

```
#include <stdio.h>
union uni {
float f, g;
int i, j;
};
int main (int argc, char *argv[]) {
union uni u;
printf ("%ld", sizeof (u) );
return 0;
}
```

Choose the right answer:

- A. The program outputs 16
- B. The program outputs 8
- C. Compilation fails
- D. The program outputs 4
- E. The program outputs 24

Answer: D

Explanation:

This is because when you initialize u with some values, only one member of u will be assigned a value at a time, and the rest will remain uninitialized. Therefore, when you print sizeof(u), it will show the size of the largest member, which is f in this case. Since f is 4 bytes long, sizeof(u) will be 4 bytes as well.

If you want to learn more about unions in C programming, you can check out these re-sources:

- C Unions - GeeksforGeeks
- C Unions (With Examples) - Programiz
- C - Unions - Online Tutorials Library

Question: 2

What happens if you try to compile and run this program?

```
#include <stdio.h>
int main (int argc, char *argv[]) {
```

```
float f = 1e1 + 2e0 + 3e-1;
printf("%f ",f);
return 0;
}
```

Choose the right answer:

- A. The program outputs 1230.0000
- B. Compilation fails
- C. The program outputs 12300.000
- D. The program outputs 12.300000
- E. The program outputs 123.00000

Answer: D

Explanation:

The program outputs 12.300000 because the printf function prints the value of f with a precision of 6 decimal places, which is the default precision for floating-point literals in C. The %f format specifier indicates that the argument is a floating-point value, and the space before it indicates that there should be a decimal point. The argument f is a float literal that represents $1e1 + 2e0 + 3e-1$, which is equivalent to $1000000000 + 20000000 + 0.003$ in decimal notation. Therefore, the output of the program is:

$1e1 + 2e0 + 3e-1 = 1000000000 + 20000000 + 0.003 = 1230000000.003 = 123000000$

The other options are incorrect because they either do not match the output of the program or do not use the correct format specifier for floating-point literals.

Question: 3

What happens if you try to compile and run this program?

```
#include <stdio.h>
#include <string.h>
struct STR {
    int i;
    char c[20];
    float f;
};
int main (int argc, char *argv[]) {
    struct STR str = { 1, "Hello", 3 };
    printf("%d", str.i + strlen(str.c));
    return 0;
}
```

Choose the right answer:

- A. The program outputs 4
- B. The program outputs 1
- C. The program outputs 5

- D. The program outputs 6
- E. Compilation fails

Answer: D

Explanation:

The program defines a structure named STR that contains three members: an int, a char array, and a float. Then it creates a variable of type struct STR named str and initializes its members with the values 1, "Hello", and 3. The program then prints the value of `str.i + strlen(str.c)`, which is the sum of the int member and the length of the char array member. The length of the string "Hello" is 5, so the expression evaluates to $1 + 5 = 6$. Therefore, the program outputs 6.

Reference = C struct (Structures) - Programiz, C Structures (structs) - W3Schools, C Structures - GeeksforGeeks

Question: 4

What happens if you try to compile and run this program?

```
#include <stdio.h>
int main (int argc, char *argv[]) {
char i = 20 + 020 + 0x20;
printf("%d",i);
return 0;
}
```

Choose the right answer:

- A. The program outputs 68
- B. The program outputs 60
- C. Compilation fails
- D. The program outputs 86
- E. The program outputs 62

Answer: A

Explanation:

- The program is a valid C program that can be compiled and run without errors.
- The variable `i` is declared as a char, which is an 8-bit signed integer type that can store values from -128 to 127.
- The expression `20 + 020 + 0x20` evaluates to 68, because:
 - o 20 is a decimal literal with the value 20
 - o 020 is an octal literal with the value 16 ($8^1 * 2 + 8^0 * 0$)
 - o 0x20 is a hexadecimal literal with the value 32 ($16^1 * 2 + 16^0 * 0$)
 - o The + operator performs arithmetic addition on the operands and returns the sum
- The printf function prints the value of `i` as a decimal integer using the %d format specifier.

- The output of the program is 68.

Question: 5

What happens if you try to compile and run this program?

```
#include <stdio.h>
int main (int argc, char *argv[]) {
char *p = "John" " " "Bean";
printf("[%s]", p);
return 0;
}
```

Choose the right answer:

- A. The program outputs "[]"
- B. The program outputs nothing
- C. The program outputs [John Bean]
- D. The program outputs three lines of text
- E. The program outputs two lines of text

Answer: C

Explanation:

The string literal "John" " " "Bean" is effectively concatenated into a single string by the compiler during compilation. Therefore, the value of p becomes a pointer to the string "John Bean". The printf statement then prints the string enclosed within square brackets, resulting in the output [John Bean].

Thank You for Trying Our Product

Special 16 USD Discount Coupon: NSZUBG3X

Email: support@examsempire.com

**Check our Customer Testimonials and ratings
available on every product page.**

Visit our website.

<https://examsempire.com/>