Learn and Master C Programming For Absolute Beginners!

## Section 2: Quick Intro To Computer Basics

Quiz 1: Computer Basics

Question 1: What does CPU stand for?

* Central Processing Unit
* Central Programming Unit
* Code Production Unit
* Command Processing Unit
* C Programming Unit

Answer: Central Processing Unit

Question 2: What part of the CPU that is responsible for performing arithmetic and logical operations?

* AUL
* CU
* ACU
* ALU
* CPU

Answer: ALU

Question 3: Programs we write, need to use memory to store its data at run-time. This memory is known as...

* ROM
* SSD
* RAM
* EPROM
* Cache Memory

Answer: RAM

Question 4: What language does the computer and CPU understand?

* C Language
* Assembly Language
* Machine Code
* Scripting Languages

Answer: Machine Code

Question 5: C Language is considered...

* A Interpreted language
* A Compiled language
* Both of the above
* None of the above

Answer: A Compiled language

Question 6: Who is the original inventor of C Language?

* James Gosling
* Bjarne Stroustrup
* Dennis Ritchie
* Brendan Eich
* Rasmus Lerdorf

Answer: Dennis Ritchie

## Section 4: Writing our First Program - A 'Hello, World!' Program in C

Quiz 2: Understanding Your First C Program

Question 1: When you write code in C, does it have to be case sensitive?

* Yes
* No

Answer: Yes

Question 2: What is the only function all C programs MUST contain?

* entry()
* MAIN()
* main()
* start()
* program()
* system()
* begin()

Answer: main()

Question 3: What do we use in C to signal the beginning and end of code blocks?

* { and }
* ( and )
* [ and ]
* BEGIN and END
* None of the above

Answer: { and }

Question 4: What do we use to end most lines of code in C?

* END
* .
* ;
* :
* ,
* “
* None of the above

Answer: ;

Question 5: Which header file is used to define input/output function prototypes and macros in C?

* stdlib.h
* math.h
* io.h
* system.h
* print.h
* stdio.h
* None of the above

Answer: stdio.h

Question 6: What character is used to make printf output move to a new line?

* \r
* \c
* \t
* \n
* None of the above

Answer: \n

## Section 5: Walkthrough: Visual Studio IDE

Quiz 3: Understanding VS (Visual Studio) IDE

Question 1: What does IDE stand for?

* Investigational Device Exemption
* Intelligent Drive Electronics
* Integrated Device Electronics
* Integrated Development Environment
* None of the above

Answer: Integrated Development Environment

Question 2: True or False: Start Page has links to recent projects and solutions

* True
* False

Answer: True

Question 3: True or False: You MUST create a separate directory for a solution in VS

* True
* False

Answer: False

Question 4: If you want to undo changes or even redo changes to your code in VS. What menu do you use?

* File Menu
* Edit Menu
* Build Menu
* Debug Menu
* Tools Menu
* None of the above

Answer: Edit Menu

Question 5: What symbol does VS use to show TAB key when you click on "View White Space"?

* Dot “.”
* Arrow “->”
* TAB “\t”
* White Space “ “
* None of the above

Answer: Arrow “->”

Question 6: Is it better to insert TABS or SPACES into your code files?

* SPACES are better
* TABS are better
* It doesn’t matter

Answer: SPACES are better

Question 7: To quickly find some text in your code, you can type the shortcut...

* CTRL+SHIFT+F
* SHIFT+F
* CTRL+F
* CTRL+H
* SHIFT+H
* None of the above

Answer: CTRL+F

Question 8: How many projects can you have in a single solution in Visual Studio?

* One and only one
* A maximum of 2 projects
* A maximum of 3 projects
* As many as you want

Answer: As many as you want

## Section 6: Fundamentals and Basics

Quiz 4: Understand Basic Concepts about C and its Syntax

Question 1: Which of the following is a correct comment in C?

* \*\* This is a Comment\*\*
* \*/ This is a Comment\*/
* /\* This is a Comment\*/
* < This is a Comment>
* ## This is a Comment

Answer: /\* This is a Comment\*/

Question 2: A single line comment in C begins with...

* ! This is a single line comment
* @ This is a single line comment
* // This is a single line comment
* \\ This is a single line comment
* <? This is a single line comment?>
* None of the above

Answer: // This is a single line comment

Question 3: Which of the following is a keyword in C?

* auto
* break
* continue
* for
* while
* sizeof
* static
* short
* case
* begin
* All of the above
* All except “ short”
* All except “ begin”
* None of the above

Answer: All except “begin”

Question 4: Which of the following is a valid variable name in C?

* $myvar
* 123MyVar
* My Var
* My-Var
* MyVar

Answer: MyVar

Question 5: sizeof operator is used to calculate the memory size in ....

* Number of integers
* Number of Characters
* Number of Bytes
* Number of Bits

Answer: Number of Bytes

Question 6: Which of the following is NOT a valid type in C?

* real
* double
* int
* long
* char
* All of the above

Answer: real

Question 7: Which of the following printf format specifiers is used to print out integers?

* %d
* %ld
* %u
* %lu
* All of the above
* None of the above

Answer: All of the above

Question 8: How to declare an integer called "myAge" in C and initialize it to 32?

* double myAge == 32;
* int myAge == ;
* integer myAge = 32;
* int myAge = 32;
* int myAge == 32;
* myAge int = 32;

Answer: int myAge = 32;

Question 9: Is the following code valid or not?

1] const int age = 30;

2] printf("My age = %d\n", age);

3] age += 10;

* Not valid because you cannot modify a constant
* Yes, this code is valid

Answer: Not valid because you cannot modify a constant

Question 10: How to declare a new enumeration for the months of the year in C?

* int months:={“Jan”, “Feb”, “March”, ….};
* int [] months = {JAN, FEB, MARCH, APRIL, …};
* enum Months { Jan, Feb, March, April, …..};
* #define Months = enum{ Jan, Feb, March, April, ….};

Answer: enum Months { Jan, Feb, March, April, …..};

Question 11:

Is this code correct in C? What's the output?

int count = 10;

void main()

{

int count = 100;

printf("Count = %d\n", count++);

}

* This is not valid since count is defined twice. You get a compilation error
* This is valid code. Output is 10
* This is valid code. Output is 11
* This is valid code. Output is 100
* This is valid code. Output is 110
* This is valid code. Output is 111

Answer: This is valid code. Output is 100

Question 12: What is the return type of the following function?

int Foo(char \* pstr, float data);

* float
* void
* char
* char\*
* int
* long

Answer: int

## Section 7: Operators & Expressions

Quiz 5: Understanding Expressions in C

Question 1: Which of the following is the correct operator to compare two variables?

* isequal
* equal
* compare
* =
* :=
* ==

Answer: ==

Question 2: Which of the following is the logical OR operator?

* OR
* |
* ||
* &
* &&

Answer: ||

Question 3: What's the value of this expression?

int x = (2\*3+2) \* 10 - 5;

* 70
* 80
* 21
* 75
* 95
* 40

Answer: 75

Question 4: Evaluate the following expression:

(10 && 0) || 1;

* This evaluates to TRUE
* This evaluates to FALSE
* Invalid Expression

Answer: This evaluates to TRUE

Question 5: What's the value of this expression?

int x = 10;

int y = 2;

x \*=y;

x += 5\*2;

y = x--;

x %=2;

* x=9,y=29
* x=2,y=30
* x=1,y=30
* x=29,y=31
* x=29,y=30
* None of the above

Answer: x=1,y=30

Question 6: What's the output of this program?

int x = 10;

int y = 4;

int z = x / y;

float f = x / y;

float ff = (float)x / y;

printf("z=%d f=%f ff=%f\n", z, f, ff);

* z=2 f=2.0 ff=2.5
* z=2 f=2.0 ff=2.5
* z=2.5 f=2.5 ff=2.5
* z=2.5 f=2.5 ff=2.5
* None of the above

Answer: z=2 f=2.0 ff=2.5

## Section 8: Conditional Statements

Quiz 6: Understanding Conditional Statements in C

Question 1: Which of the following is a valid if...else statement in C?

Assuming:

int a = 10;

int b = 20

· if ( a > b) then

b=a;

else

a=b;

· if ( a greater than b)

b=a;

else

a=b;

· if ( a > b)

b=a;

else

a=b;

endif

· if ( a > b)

b=a;

else

a=b;

· if ( a > b)

b=a

else

a=b

Answer: if ( a > b)

b=a;

else

a=b;

Question 2: What's the output of the following program?

void main()

{

int a = 10;

int b = 20;

if (a > b)

printf("a is greater than b!\n");

printf("a=%d, b=%d\n", a, b);

printf("Bye!\n");

}

· a is greater than b!

a=10, b=20

Bye!

· a=10, b=20

Bye!

· a is greater than b!

a=10, b=20

· Bye!

· a is greater than b!

· a=10, b=20

· None of the above

Answer: a=10, b=20

Bye!

Question 3: Which of the following is the correct function to return a minimum of 2 integers?

· int min(a, b)

{

return (a < b) ? b : a;

}

· int min(a, b)

{

(a < b) ? return a : return b;

}

· int min(a, b)

{

return (a < b) ? a : b;

}

· int min(a, b)

{

if (a < b) ? return a : return b;

}

· None of the above

Answer: int min(a, b)

{

return (a < b) ? a : b;

}

Question 4: what's the output of this program?

void main()

{

int a = 3;

switch (a)

{

case 3:

printf("a=3!\n");

case 2:

printf("a=2!\n");

case 1:

printf("a=1!\n");

break;

default:

printf("Default!\n");

}

}

· a=3!

· a=3!

a=2!

a=1!

Default!

· a=3!

a=2!

a=1!

· 3!

Default!

· None of the above

Answer: a=3!

a=2!

a=1!

## Section 9: Loops, Jump keywords & Control Flow

Quiz 7: Understanding Loops and Jump Statements in C

Question 1: The following program will output...?

for (int i = 0; i > 3; ++i)

printf("%d\n", i);

• 0

• 1

• 2

• 0

• 1

• 2

• 3

• 1

• 2

• 3

● No Output

● None of the above

Answer: No Output

Question 2: Given the following program, which of the statements below let you exit the loop when 'i' is greater than 50?

for (int =0; i < 100; ++i)

{

printf("%d\n", i);

}

● return if (i > 50)

● (i>50) ? return;

● if (i > 50)

quit;

● if (i > 50)

break;

● if (i > 50)

leave;

● if (i > 50)

goto break;

Answer: if (i > 50)

leave;

Question 3: Which of the following loops is going to execute at least ONCE?

int i=10;

for (; i < 3; ++i) ....

while (i < 3) ...

do { ...} while ( i < 3);

● The 'for' loop

● The 'while' loop

● The 'do...while' loop

● All of the above

● None of the above

Answer: The 'do...while' loop

Question 4: Which of the following is an infinite loop?

for ( ; ; )

{

}

while (0)

{

}

do

{

}

while ( 0 )

● All of them

● None of them

● The 'for' loop only

● The while loops, but not the for loop

● All except the 'while' loop

Answer: The 'for' loop only

Question 5: Which of the following keywords below can be used inside loops?

break;

continue;

goto;

if

switch

● All except 'goto'

● All except 'continue'

● All except 'if'

● All of them

● None of them

● all except switch

Answer: All of them

## Section 10: Arrays

Quiz 8: Understanding Arrays in C

Question 1: How to declare one dimensional array of 10 elements in C?

● int [] a = new int[10];

● int a[] = int[10]

● int a[10];

● a[10] int;

● int\* a[10];

● None of the above

Answer: int a[10];

Question 2: Which of the following function declarations can be considered similar?

int MyFunction(int a[]);

int MyFunction2(int\* a);

int MyFunction3(int\* a[]);

● All of them are equivalent (similar)

● MyFunction and MyFunction2 are similar

● They're all completely different

Answer: MyFunction and MyFunction2 are similar

Question 3: What's the output of this code assuming integer is 4 bytes?

void main()

{

int arr[] = { 1, 2, 3, 4, 5, 6 };

printf("%d\n", sizeof(arr));

printf("%d\n", sizeof(arr[0]));

}

6

4

24

6

24

4

● None of the above

Answer: 24

4

Question 4: How to declare 2 dimensional array in C?

● int [][] a = new int[3][2];

● int [3][2]a;

● a[3][2] int;

● int a[3][2];

● int a[3,2];

● int a[3]a[2];

Answer: int a[3][2];

Question 5: What's the output of this program?

void main()

{

char s1[] = { 'a', 'b', 'c', 'd' };

char s2[] = "abcd";

printf("%d, %d\n", sizeof(s1), sizeof(s2));

}

● 4,4

● 4,5

● 5,5

● 16,16

● 16,20

Answer: 4,5

## Section 11: Pointers

Quiz 9: Understanding Pointers in C

Question 1: What operator is used to give us the address of a variable?

● &

● \*

● @

● #

● $

Answer: &

Question 2: What is the output of this program?

void main()

{

int a = 100;

&a = 200;

printf("%d\n", a);

}

● 100

● 200

● Compilation Error

● None of the above

Answer: Compilation Error

Question 3: What's the output of this program?

void foo(int a, int b, int c)

{

int temp = a;

a = b;

b = c;

c = temp;

}

void main()

{

int a = 1, b = 2, c = 3;

foo(c, b, a);

printf("%d, %d, %d\n", a, b, c);

}

● 2, 1, 3

● 3, 2, 1

● 1, 2, 3

● 1, 3, 2

Answer: 1, 2, 3

Question 4: What's wrong with this code?

void main()

{

int\* p;

\*p = 100;

printf("%d\n", \*p);

}

● Nothing wrong. It prints out 100

● Uninitialized local variable 'p' used. This could cause the program to crash because we're modifying some unknown place in memory!

● This code will not compile

● printf should use &p instead of \*p

Answer: Uninitialized local variable 'p' used. This could cause the program to crash because we're modifying some unknown place in memory!

Question 5: What's the output of this program?

void main()

{

int a = 123;

int\* p = &a;

int b = \*p;

b++;

\*p++;

p = &a;

printf("%d, %d, %d", a, b, \*p);

}

● Compilation Errors

● Program will crash

● 123, 124, 125

● 125, 125, 125

● 123, 125, 123

● 124, 124, 124

● 123, 124, 123

● 123, 123, 123

● None of the above

Answer: 123, 124, 123

Question 6: What's wrong with this code?

void main()

{

const char \* p = "one Two Three";

p[0] = "O";

p[1] = "N"

p[2] = "E";

}

● You cannot modify a constant string (read-only memory)

● This is valid code. the first word becomes "ONE"

● Compilation Erros

Answer: You cannot modify a constant string (read-only memory)

Question 7: what's the output of this code?

void main()

{

const char \* p = "Coding is Fun!";

printf(p + 10);

}

● Coding is Fun!10

● Fun!

● Compilation Error

● None of the above

Answer: Fun!

Question 8:

void main()

{

const char \* p = "Coding is Fun!";

p+=10;

printf(p);

}

● Fun!

● Compilation Error

● Nothing

Answer: Fun!

Question 9: What's the output of this C code?

void main()

{

int a[3] = {10, 20, 30};

int \*p = a;

printf("%p\t%p", p, a);

}

● Same address is printed out

● Compilation Error

● Different address is printed out

Answer: Same address is printed out

Question 10: Which of the following is a constant pointer?

void main()

{

int a = 10;

int b = 20;

const int c = 30;

int \* const p1 = &a;

const int \* p2 = &c;

const int \* const p3 = &c;

int const \* p4 = &b;

}

● All of them

● p2, p3

● p2, p3, p4

● p1, p3

● p1

● p3

● p4

Answer: p1, p3

Question 11: What is the output of this code?

void main()

{

printf("%d\n", sizeof(void\*));

}

● Compilation error, because void doesn't have a size

● The size of a pointer in bytes. It will be 4 bytes on 32-bit system and 8 bytes on 64-bit

● 2 bytes

● Always 4 bytes

● None of the above

Answer: The size of a pointer in bytes. It will be 4 bytes on 32-bit system and 8 bytes on 64-bit

## Section 12: Strings

Quiz 10: Understanding Strings in C

Question 1: Assuming pointers are 4 bytes, what is the output of this code?

printf("%d\n", sizeof("Fun with Code!"));

• 4

• 8

• 14

• 15

• None of the above

Answer: 15

Question 2: What's the output of this C code?

1. void main()

2. {

3. char str[] = "Weclome!";

4. const char \* p = str;

5. printf("%c%c", str[0], p[1]);

6. }

• Compile Error

• Run time Error

• Program will crash

• WW

• We

• Wc

Answer: We

Question 3:

What does this program do?

1. void main()

2. {

3. char s[] = "Welcome to this quiz!\n";

4. char \* p = &s[0];

5. do

6. {

7. if (\*p >= 'a' && \*p <= 'z')

8. printf("%c", (\*p) - 'a'+'A' );

9. else

10. printf("%c", \*p);

11.

12. } while (\*p++);

13. }

• Compile Error

• Run-time Error

• WELCOME TO THIS QUIZ!

• Program will crash

• Welcome to this quiz!

• None of the above

Answer: WELCOME TO THIS QUIZ!

Question 4: What does the following function do?

1. int foo(const char \* p)

2. {

3. if (NULL == p || !\*p)

4. return 0;

5. int count = 0;

6. while (\*p++)

7. count++;

8. return count;

9. }

• Counts the number of characters in a string like strlen

• Counts the size of a string in bytes

• Counts the number of NULL characters in a string

• Counts and increments all characters of a string

• None of the above

Answer: Counts the number of characters in a string like strlen

Question 5: What does the following code print out?

1. void main()

2. {

3. const char \* p1 = "Hi there!";

4. const char \* p2 = "How is your day so far?"

5. char str[240];

6.

7. strcpy(str, p1);

8. strcpy(str, p2);

9. printf(str);

10. }

• Hi there!

• Hi there!How is your day so far?

• Hi there! How is your day so far?

• How is your day so far?

• Run time error

• None of the above

Answer: How is your day so far?

Question 6:

What's an easy way to convert a number to string in C?

Assuming, we have:

1. int a = 123;

2. char str[100];

• strcpy(str, a);

• strcat(str, a);

• str=a;

• sprintf(str, "%d", a);

• All of the above

• None of the above

Answer: sprintf(str, "%d", a);

Question 7:

Is this code valid or not?

1. char str[100];

2.

3. str="C is cool!";

• No, this is not valid. You cannot assign a constant string like this. You must use: strcpy(str, "C is cool");

• Yes, this is perfectly fine!

Answer: No, this is not valid. You cannot assign a constant string like this. You must use: strcpy(str, "C is cool");

Question 8:

Which of the following is used to compare two strings?

1. sprintf

2. strcpy

3. strcat

4. strcmp

5. strstr

• All of the above

• strstr

• strcmp

• strcat

• strcpy

Answer: strcmp

Question 9:

How do we pass arguments to main function?

1. 1. void main(int argc, const char \*\* argv);

2. 2. void main(const int argc, const char \*\* argv);

3. 3. void main(int argc, char \* argv[]);

4. 4. void main(int argc, const char \* argv[]);

5. 5. int main( int argc, char \*\* argv);

6. 6. int main( char \* argv[], int argc);

• 1

• 4

• 6

• 1 through 6

• 1 through 5, but not 6

• None of the above

Answer: 1 through 5, but not 6

## Section 14: Preprocessor Directives

Quiz 11: Using Preprocessor Directives in C

Question 1: What's the result of using this macro in this code example?

• #define ADD( a, b) a+b

•

• void main()

• {

• printf("%d\n", ADD(2,3)\*2);

• }

• 10

• 8

• 6

• Compile Error

Answer: 8

Question 2:

Which of the following #include directives looks for the file in the same directory as the file including it as a first step?

1. #include <myfile.h>

2. #include "myfile.h"

• 2

• 1

• Both of them

• There is no way for us to tell which directory it starts looking at

Answer: 2

Question 3: What is the output after running this code?

1. #define RELEASE

2.

3. void main()

4. {

5. #if defined(LINUX)

6. printf("LINUX defined");

7. #if defined(RELEASE)

8. printf("Release defined");

9. #endif

10. #endif

11. }

• 1. Linux defined

2. Release defined

• 1. Linux defined

• Nothing

• Compile Errors

Answer: Nothing

## Section 15: Working with Header Files and Multiple Source Files

Quiz 12

Question 1: Why do we use #pragma once in our code?

• To prevent the compiler from compiling our code

• To avoid multiple inclusion problem

• All of the above

• None of the above

Answer: To avoid multiple inclusion problem

## Section 16: Creating C Libraries ( Static LIBs)

Quiz 13: Working with static libraries in C

Question 1: One of the ways to link with my.lib is to use...

• #pragma lib("mylib")

• #pragma "mylib"

• #pragma comment(lib, "mylib")

• #pragma lib(comment, "mylib")

Answer: #pragma comment (lib, "mylib")

Top of Form

Question 2:

What's missing from this code?

1. void main()

2. {

3. SayHello();

4. }

5.

6. void SayHello()

7. {

8. printf("Hello");

9. }

• Nothing is missing. This code works and compile with no warnings or errors

• Program will crash at run-time

• We should forward declare SayHello before we use it.

Answer: We should forward declare SayHello before we use it.

Question 3: When you break your code into multiple source files (.c), Can you call functions in other source files?

• No. You can only call local functions in the same file.

• Yes, it should just work!

• Yes, but you need to forward declare any function(s) that you plan to call from other files.

• You cannot break your code into multiple files

Answer: Yes, but you need to forward declare any function(s) that you plan to call from other files.

## Section 17: Structures & Unions

Quiz 14: Structs and Unions Quiz

Question 1: What's missing from the following code?

1. struct Rect{

2. int left;

3. int right;

4. int top;

5. int bottom;

6. };

7.

8. void main()

9. {

10. Rect rc = { 10, 20, 30, 40};

11. }

• Compilation Error because in C, you need to declare a struct with the keyword 'struct' like:

struct Rect rc = {10, 20, 30 ,40};

• Nothing is missing. This should work just fine

Answer: A binary file with .LIB extension that you cannot run directly.

Question 2: Assuming int is 4 bytes, which of the following is true about structs and unions?

1. struct stRect{

2. int left;

3. int right;

4. int top;

5. int bottom;

6. };

7.

8. union unRect {

9. int left;

10. int right;

11. int top;

12. int bottom;

13. };

1. Both have members that can be individually accessed using either "." or "->" notation

2. Both are UDDT (User Defined Data Types)

3. sizeof( struct stRect) == sizeof(union unRect);

• 1,2 and 3 are all true

• 1,2 only because sizeof struct is 16 where as sizeof union is 4

• Only 2 is true

• Only 1 is true

Answer: 1, 2 only because sizeof struct is 16 where as sizeof union is 4

## Section 18: Working with Files and I/O

Quiz 14: Working with Files in C

Question 1: Which is True?

* Text files are special type of Binary files
* Binary files are a special type of text files

Answer: Option 1

Question 2: fopen can write to a file if the given mode is:

1. "r"

2. "w"

3. "a"

4. "r+"

5. "w+"

6. "a+"

* Only #2
* Only #2 and #5
* All of the above (#1 to #6)
* All except #1 (#2 to #6)
* #2, #3, #5, #6

Answer: All except #1 (#2 to #6)

Question 3: Which of the following can be used to read 1 character from a text file?

1. fgetc

2. fread

3. fscanf

4. fgets

* Only #1 (fgetc)
* #1 and #2
* #1, #2 and #3 but not #4
* All of the above

Answer: All of the above

Question 4: If we successfully fopen a file, we must close it using:

1. fclose

2. feof

3. DeleteFile

4. close

* #1
* #1 and #2
* #2
* #3
* #4

Answer: #1

Question 5: which function can be used to read a single line of text in one call?

1. fgetc

2. fputc

3. fputs

4. fgets

5. fread

* #1
* #2
* #3
* #4
* #5
* All of the above

Answer: #4

## Section 19: Let's Talk Binary | Manipulating Bits

Quiz 15: Bit manipulation in C

Question 1: What's the output of this program?

int a = 3;

int b = 4;

printf("%d\n", a & b);

printf("%d\n", a && b)

* True
* True

1. 7
2. 7

* 0
* 1
* 7
* 1
* 7
* -1

Answer:

* 0
* 1

Question 2: What's the output for this program?

int a = 3;

int b = 4;

printf("%d\n", a | b);

printf("%d\n", a || b);

* -1
* 7
* 7
* 1
* 0
* 1
* 1
* 1
* 12
* 1

Answer:

 7

 1

Question 3: Predict the output of this program in C:

void main()

{

int a = 12;

if (a & 4)

a &= ~4;

printf("%d\n", a);

}

* 12
* 4
* -4
* 8
* 3
* 0
* 1

Answer: 8

Question 4: What's the output of this C code?

void main()

{

int a = 12;

int b = a >> 2;

int c = a << 1;

printf("%d, %d, %d\n", a, b, c);

}

* 12, 0, 2
* 12, 10, 13
* 12, 3, 24
* 12, 6, 12
* None of the above

Answer: 12, 3, 24

## Section 20: Dynamic Memory Allocation

Quiz 16: Understanding Dynamic Memory Management in C

Question 1: Which is larger in size: Stack or Heap memory?

* Both are the same size
* Heap is larger and allows for larger memory allocations
* Stack is larger and faster than heap
* None of the above

Answer: Heap is larger and allows for larger memory allocations

Question 2: If you want to process data files that have different sizes (could be 100 of megabytes), it is better to use...

* The stack memory, because it is faster to process and the code will be easier to write
* The heap memory, because placing large objects on the stack may cause stack overflow

Answer: The heap memory, because placing large objects on the stack may cause stack overflow

Question 3: What’s the problem with this code?

Void main()

{

void \* p = malloc(1024);

memset(p, ‘a’, 1024);

printf(“%c\n”, \*(char\*)p);

}

* You cannot use void \* with characters
* You cannot use void \* with malloc
* The program will crash
* The code will print out ‘a’
* We’re causing a memory leak here! Even though this code will print out ‘a’ on the screen, we should free the heap memory calling by free(p)
* Compile error

Answer: We’re causing a memory leak here! Even though this code will print out ‘a’ on the screen, we should free the heap memory calling by free(p)

## Section 20: More C Keywords...

Quiz 18: More C Keywords – Quiz

Question 1: Predict the output of this program:

1. static float count = 4.0f;

2. void main()

3. {

4. auto int a = 10;

5. int b = 20;

6.

7. typedef int Integer;

8.

9. Integer count = 2;

10.

11. for (register Integer i = 0; i < count; ++i)

12. printf("%d\n", i+b-a);

13.

14.}

Option 1:

Compile Errors:

* Line 1: count is defined twice with different types
* Line 7: typedef cannot be inside the main function
* Line 9: Invalid Integer
* Line 11: register is not allowed here

Option 2:

* 10
* 11

Answer: Option 2

Question 2:

static float count = 4.0f;

void main()

{

int count = 2;

printf("%d\n", count);

}

What's the output here?

* 2
* 4.0

Answer: 2

Question 3:

void main()

{

typedef int Integer;

Integer count = 2;

}

* This code will not compile
* Integer becomes a new type definition and cannot be use as an integer

Answer: Integer becomes a new type definition and cannot be use as an integer

Question 4:

void main()

{

int count = 2;

for (register int i = 0; i < count; ++i)

printf("%d\n", i+b-a);

}

* Register forces i to be placed inside a CPU register for faster performance
* Register is a suggestion (hint) to the compiler to consider using a CPU register for the loop counter “i”, but it may actually ignore this recommendation
* Compilation errors due to invalid use of “ register” keyword

Answer: Register is a suggestion (hint) to the compiler to consider using a CPU register for the loop counter “i”, but it may actually ignore this recommendation

## Section 22: Advanced Topics

Quiz 19: Advanced Topics Quiz

Question 1: What's wrong with this recursive function?

int print(int i)

{

return print (i -1);

}

* Compilation Errors
* This is valid and returns i – 1. For example print(2) returns 1
* This recursive function will run infinitely (has no termination condition) and will cause Stack Overflow at runtime!

Answer: This recursive function will run infinitely (has no termination condition) and will cause Stack Overflow at runtime!

Question 2: Predict the output here:

void print(int i)

{

printf("%d\n", i);

if (i < 2)

return ;

print(i/2);

}

void main()

{

print(10);

}

* Stack Overflow
* 10
* 10 and 2
* 10, 5, 2, 1
* 10, 5, 2

Answer: 10, 5, 2, 1

Question 3: Which of the following functions is a variadic function?

1. printf(const char \* p, ...);

2. void Foo(...);

3. void Count(int a, ...);

4. int Max(a[]);

5. int Min(int \* pa);

* Only #1
* All except #2
* #1 and #3 only

Answer: #1 and #3 only

## Section 23: Mixing C with C++ Code

Quiz 20: Mixing C/C++ Quiz

Question 1: Which compiler has \_\_cplusplus defined?

* C++ Compiler Only
* C Compiler Only
* Both C and C++ Compilers

Answer: C++ Compiler Only

Question 2: When and where do we use extern "C"?

* We use it with our C++ headers if we want to include them in our C Code
* We use them on our C headers when we want to use them in C++ code
* We should use them on all C/C++ functions

Answer: We use them on our C headers when we want to use them in C++ code