Read c book for following questions.

Q1. The default calling convention for the C program is \_\_\_\_\_\_.

\*1. cdecl\*

2. pascal

3. stdcall

4. None of above.

02. cdecl stands for \_\_\_\_\_\_\_\_\_.

1. c declarator

\*2. c declaration\*

3. c programming declarator

4. none of above

03. in cdecl calling convention argument push order is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

\*1.right to left\*

2. left to right

3. right to middle and middle to left

4. none of above

04.in pascal calling convention argument push order is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

1.right to left

\*2. left to right\*

3. right to middle and middle to left

4. none of above

05.in stdcall calling convention argument push order is \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

\*1.right to left\*

2. left to right

3. right to middle and middle to left

4. none of above

06.in cdecl calling convention stack cleanup is done by      \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

\*1. calling function\*

2. called function

3. depends on compliler

4. none of above

07. in pascal calling convention stack cleanup is done by      \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

1. calling function

\*2. called function\*

3. depends on compliler

4. none of above

08. in stdcall calling convention stack cleanup is done by      \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

1. calling function

\*2. called function\*

3. depends on compliler

4. none of above

09. The conditional compilation directives are

1. #if, #else, #elif

2. #ifdef, #ifndef, #endif

3. #undef

\*4. all of above\*

10.  \_\_\_\_\_\_\_ is a preprocessor operator, which converts a macro argument to string constant

\*1. #\*

2. ##

3. #include

4. #define

11. \_\_\_\_\_\_\_\_\_  is a preprocessor operator, which is used to combine two tokens together.

1. #

\*2. ##\*

3. #include

4. #define

12. \_\_\_\_\_\_ is a special purpose directive that can be used to turn ON or OFF certain features of compiler and linker.

\*1. #pragma\*

2. #define

3. #include

4. all of above.

13. Text section also referred as \_\_\_\_\_\_\_\_\_\_\_\_.

\*1. Code section\*

2. data section

3. symbol table

4. Exe Header

5. stack section

14. \_\_\_\_\_\_\_\_\_\_ section contains all global and static variables and there initial values.

1. Code

\*2. data\*

3. symbol table

4. Exe Header

5. stack

15. \_\_\_\_\_\_\_\_\_\_ section also contains magic number that is a specific number representing the

operating system on which the program can be executed.

1. Code

2. data

3. symbol table

\*4. Exe Header\*

5. stack

16. \_\_\_\_\_\_\_\_\_\_\_section contains the type of executable file.

1. Code

2. data

3. symbol table

\*4. Exe Header\*

5. stack

17. \_\_\_\_\_\_\_\_\_\_ section includes address of entry point function i.e. main().

1. Code

2. data

3. symbol table

\*4. Exe Header\*

5. stack

18. \_\_\_\_\_\_\_\_ section is created at run time as a part of process memory.

1. Code

2. data

3. symbol table

4. Exe Header

\*5. stack\*

19. default storage is \_\_\_\_\_\_\_\_\_\_.

\*1. local or automatic\*

2. register

3. global or extern

4. static

20. local or automatic variables created on \_\_\_\_\_\_\_\_\_\_.

1. Code section

2. data section

3. cpu

\*4. stack section\*

21. register variables taken from \_\_\_\_\_\_\_\_\_\_\_.

1. Code section

2. data section

\*3. cpu\*

4. stack section

22. global and static variables stored on \_\_\_\_\_\_\_\_\_.

1. Code section

\*2. data section\*

3. cpu

4. stack section

23. #include <stdio.h>

int main()

{

    printf("\n  %d", 0100);

    return 0;

}

1. 100

2. 0100

\*3. 64\*

4. 36

24. #include <stdio.h>

int main()

{

    int no1=(2,3,4,5);

    int no2=(++no1, no1++);

    printf("\n no1= %d no2= %d", no1, no2);

    return 0;

}

1. no1= 7 no2= 7

\*2. no1= 7 no2= 6\*

3. no1= 6 no2= 7

4. no1= 6 no2= 6

25. #include <stdio.h>

int main()

{

    int no1=(2,3,4,5);

    int no2=(++no1, no1++);

    no1= 2,4,6,8;

    no2= ++no1, no1++;

    printf("\n no1= %d no2= %d", no1, no2);

    return 0;

}

\*1.  no1= 4 no2= 3\*

2.  no1= 10 no2= 9

3.  no1= 10 no2= 10

4.  no1= 4 no2= 4

26. #include <stdio.h>

int main()

{

    int no1=2,3,4,5;

    printf("\n no1= %d ", no1);

    return 0;

}

1.  no1= 2

2.  no1= 5

\*3.  compile time error\*

4.  run time error

27. \_\_\_\_\_\_\_\_ function is used to convert string into integer.

\*1. atoi\*

2. atol

3. atoll

4. all of above