IndiaBIX

Search...

Home Aptitude Logical Verbal CA Current Affairs GK Engineering Interview Online Test Puzzles

Online C Programming Test :: C Programming Test 6

Home » Online Test » Online C Programming Test » C Programming Test 6

Marks : 14/20

Total number of questions	:	**20**
Number of answered questions	:	**20**
Number of unanswered questions	:	**0**

Test Review : View answers and explanation for this test.

1.What will be the output of the program, if a *short int* is 2 bytes wide?

```
#include<stdio.h>
int main()
{
    short int i = 0;
    for(i<=5 && i>=-1; ++i; i>0)
        printf("%u,", i);
    return 0;
}
```

☑ A.1 ... 65535 ✓
☐ B.Expression syntax error ✗
☐ C.No output ✗
☐ D.0, 1, 2, 3, 4, 5 ✗

Your Answer: Option A

Correct Answer: Option A

Explanation:

for(i<=5 && i>=-1; ++i; i>0) so expression *i<=5 && i>=-1* initializes *for* loop. expression *++i* is the loop condition. expression *i>0* is the increment expression.

In *for(i <= 5 && i >= -1; ++i; i>0)* expression *i<=5 && i>=-1* evaluates to one.

Loop condition always get evaluated to *true*. Also at this point it increases *i* by one.

An increment_expression *i>0* has no effect on value of *i*.so for loop get executed till the limit of integer (ie. 65535)

Learn more problems on : Control Instructions

Discuss about this problem : Discuss in Forum

2.What will be the output of the program?

```
#include<stdio.h>
int main()
{
    float a = 0.7;
    if(0.7 > a)
        printf("Hi\n");
    else
        printf("Hello\n");
    return 0;
}
```

☑ A.Hi ✓

☐ B.Hello ✗

☐ C.Hi Hello ✗

☐ D.None of above ✗

Your Answer: Option A

Correct Answer: Option A

Explanation:

if(0.7 > a) here *a* is a float variable and *0.7* is a double constant. The double constant *0.7* is greater than the float variable *a*. Hence the *if* condition is satisfied and it prints *'Hi'*

Example:

```
#include<stdio.h>
int main()
{
    float a=0.7;
    printf("%.10f %.10f\n",0.7, a);
    return 0;
}
```

Output:

```
0.7000000000 0.6999999881
```

Learn more problems on : Control Instructions

Discuss about this problem : Discuss in Forum

3. Which of the following statements are correct about an *if-else* statements in a C-program?

1: Every *if-else* statement can be replaced by an equivalent statements using *?:* operators

2: Nested *if-else* statements are allowed.

3: Multiple statements in an *if* block are allowed.

4: Multiple statements in an *else* block are allowed.

☑ A.1 and 2 ✗

☐ B.2 and 3 ✗

☐ C.1, 2 and 4 ✗

☐ D.2, 3, 4 ✓

Your Answer: Option A

Correct Answer: Option D

Learn more problems on : Control Instructions

Discuss about this problem : Discuss in Forum

4. Can we use a *switch* statement to switch on strings?

☐ A. Yes ✗
☑ B. No ✓

Your Answer: Option B

Correct Answer: Option B

Explanation:

The cases in a *switch* must either have integer constants or constant expressions.

Learn more problems on : Control Instructions

Discuss about this problem : Discuss in Forum

5.What will be the output of the program?

```
#include<stdio.h>
#include<math.h>
int main()
{
    printf("%f\n", sqrt(36.0));
    return 0;
}
```

☐ A. 6.0 ✗
☐ B. 6 ✗
☐ C. 6.000000 ✓
☑ D. Error: Prototype *sqrt()* not found. ✗

Your Answer: Option D

Correct Answer: Option C

Explanation:

printf("%f\n", sqrt(36.0)); It prints the square root of 36 in the float format(i.e 6.000000).

Declaration Syntax: *double sqrt(double x)* calculates and return the positive square root of the given number.

Learn more problems on : Floating Point Issues

Discuss about this problem : Discuss in Forum

6.Is it true that too many recursive calls may result into stack overflow?

☑ A. Yes ✓
☐ B. No ✗

Your Answer: Option A

Correct Answer: Option A

Explanation:

Yes, too many recursive calls may result into *stack overflow*. because when a function is called its return address is stored in stack.

After sometime the stack memory will be filled completely. Hence *stack overflow* error will occur.

Learn more problems on : Functions

Discuss about this problem : Discuss in Forum

7. A macro must always be defined in capital letters.

□ A. True ✗
☑ B. False ✓

Your Answer: Option B

Correct Answer: Option B

Explanation:

FALSE, The macro is case insensitive.

Learn more problems on : C Preprocessor

Discuss about this problem : Discuss in Forum

8. In a macro call the control is passed to the macro.

☑ A. True ✗
□ B. False ✓

Your Answer: Option A

Correct Answer: Option B

Explanation:

False, Always the macro is substituted by the given text/expression.

Learn more problems on : C Preprocessor

Discuss about this problem : Discuss in Forum

9. A header file contains macros, structure declaration and function prototypes.

☑ A. True ✓
□ B. False ✗

Your Answer: Option A

Correct Answer: Option A

Explanation:

True, the header file contains classes, function prototypes, structure declaration, macros.

Learn more problems on : C Preprocessor

Discuss about this problem : Discuss in Forum

10. Which of the following function sets first n characters of a string to a given character?

☐ A.strinit() ✗
☑ B.strnset() ✓
☐ C.strset() ✗
☐ D.strcset() ✗

Your Answer: Option B

Correct Answer: Option B

Explanation:

Declaration:

*char *strnset(char *s, int ch, size_t n);* Sets the first *n* characters of *s* to *ch*

```
#include <stdio.h>
#include <string.h>

int main(void)
{
   char *string = "abcdefghijklmnopqrstuvwxyz";
   char letter = 'x';

   printf("string before strnset: %s\n", string);
   strnset(string, letter, 13);
   printf("string after  strnset: %s\n", string);

   return 0;
}
```

Output:

string before strnset: abcdefghijklmnopqrstuvwxyz

string after strnset: xxxxxxxxxxxxxnopqrstuvwxyz

Learn more problems on : Strings

Discuss about this problem : Discuss in Forum

11. What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    char str1[] = "Hello";
    char str2[10];
    char *t, *s;
    s = str1;
    t = str2;
    while(*t=*s)
        *t++ = *s++;
    printf("%s\n", str2);
    return 0;
}
```

☐ A.Hello ✓
☐ B.HelloHello ✗
☑ C.No output ✗
☐ D.ello ✗

Your Answer: Option C

Correct Answer: Option A

Learn more problems on : Strings

Discuss about this problem : Discuss in Forum

12. What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    char str = "IndiaBIX";
    printf("%s\n", str);
    return 0;
}
```

☑ A. Error ✓
☐ B. IndiaBIX ✗
☐ C. Base address of *str* ✗
☐ D. No output ✗

Your Answer: Option A

Correct Answer: Option A

Explanation:

The line *char str = "IndiaBIX";* generates "Non portable pointer conversion" error.

To eliminate the error, we have to change the above line to

*char *str = "IndiaBIX";* (or) *char str[] = "IndiaBIX";*

Then it prints "IndiaBIX".

Learn more problems on : Strings

Discuss about this problem : Discuss in Forum

13. If the size of pointer is 4 bytes then What will be the output of the program ?

```
#include<stdio.h>

int main()
{
    char *str[] = {"Frogs", "Do", "Not", "Die", "They", "Croak!"};
    printf("%d, %d", sizeof(str), strlen(str[0]));
    return 0;
}
```

☐ A. 22, 4 ✗
☐ B. 25, 5 ✗
☑ C. 24, 5 ✓
☐ D. 20, 2 ✗

Your Answer: Option C

Correct Answer: Option C

Explanation:

Step 1: *char *str[] = {"Frogs", "Do", "Not", "Die", "They", "Croak!"};* The variable str is declared as an pointer to the array of 6 strings.

Step 2: *printf("%d, %d", sizeof(str), strlen(str[0]));*

sizeof(str) denotes 6 * 4 bytes = 24 bytes. Hence it prints '24'

strlen(str[0])); becomes *strlen(Frogs))*. Hence it prints '5';

Hence the output of the program is 24, 5

Hint: If you run the above code in 16 bit platform (Turbo C under DOS) the output will be 12, 5. Because the pointer occupies only 2 bytes. If you run the above code in Linux (32 bit platform), the output will be 24, 5 (because the size of pointer is 4 bytes).

Learn more problems on : Strings

Discuss about this problem : Discuss in Forum

14. Point out the error in the program?

```
#include<stdio.h>

int main()
{
    FILE *fp;
    fp=fopen("trial", "r");
    fseek(fp, "20", SEEK_SET);
    fclose(fp);
    return 0;
}
```

☐ A. Error: unrecognised Keyword *SEEK_SET* ✗
☑ B. Error: *fseek()* long offset value ✓
☐ C. No error ✗
☐ D. None of above ✗

Your Answer: Option B

Correct Answer: Option B

Explanation:

Instead of "20" use 20L since *fseek()* need a long offset value.

Learn more problems on : Input / Output

Discuss about this problem : Discuss in Forum

15. What will be the output of the program (myprog.c) given below if it is executed from the command line?
cmd> *myprog friday tuesday sunday*

```
/* myprog.c */
#include<stdio.h>

int main(int argc, char *argv[])
{
    printf("%c", *++argv[1]);
    return 0;
}
```

☑ A.r ✓

☐ B.f ✗

☐ C.m ✗

☐ D.y ✗

Your Answer: Option A

Correct Answer: Option A

Learn more problems on : Command Line Arguments

Discuss about this problem : Discuss in Forum

16.What will be the output of the program?

```
#include<stdio.h>

int main()
{
    const int x=5;
    const int *ptrx;
    ptrx = &x;
    *ptrx = 10;
    printf("%d\n", x);
    return 0;
}
```

☐ A.5 ✗

☐ B.10 ✗

☑ C.Error ✓

☐ D.Garbage value ✗

Your Answer: Option C

Correct Answer: Option C

Explanation:

Step 1: *const int x=5;* The constant variable *x* is declared as an integer data type and initialized with value '5'.

Step 2: *const int *ptrx;* The constant variable *ptrx* is declared as an integer pointer.

Step 3: *ptrx = &x;* The address of the constant variable *x* is assigned to integer pointer variable ptrx.

Step 4: **ptrx = 10;* Here we are indirectly trying to change the value of the constant vaiable *x*. This will result in an error.

To change the value of *const* variable *x* we have to use **(int *)&x = 10;*

Learn more problems on : Const

Discuss about this problem : Discuss in Forum

17.Point out the error in the program (in Turbo-C).

```
#include<stdio.h>
#define MAX 128
```

```
int main()
{
    const int max=128;
    char array[max];
    char string[MAX];
    array[0] = string[0] = 'A';
    printf("%c %c\n", array[0], string[0]);
    return 0;
}
```

☑ A. Error: unknown max in declaration/Constant expression required ✓

☐ B. Error: invalid array string ✖

☐ C. None of above ✖

☐ D. No error. It prints A A ✖

Your Answer: Option A

Correct Answer: Option A

Explanation:

Step 1: A macro named *MAX* is defined with value 128

Step 2: *const int max=128;* The constant variable *max* is declared as an integer data type and it is initialized with value 128.

Step 3: *char array[max];* This statement reports an error "constant expression required". Because, we cannot use variable to define the size of array.

To avoid this error, we have to declare the size of an array as static. Eg. *char array[10];* or use macro *char array[MAX];*

Note: The above program will print A A as output in Unix platform.

Learn more problems on : Const

Discuss about this problem : Discuss in Forum

18. Point out the error in the program.

```
#include<stdio.h>
const char *fun();

int main()
{
    char *ptr = fun();
    return 0;
}
const char *fun()
{
    return "Hello";
}
```

☐ A. Error: Lvalue required ✖

☑ B. Error: cannot convert '*const char* *' to '*char* *'. ✖

☐ C. No error and No output ✓

☐ D. None of above ✖

Your Answer: Option B

Correct Answer: Option C

Learn more problems on : Const

Discuss about this problem : Discuss in Forum

19.Input/output function prototypes and macros are defined in which header file?

☐ A.conio.h ✗
☐ B.stdlib.h ✗
☑ C.stdio.h ✓
☐ D.dos.h ✗

Your Answer: Option C

Correct Answer: Option C

Explanation:

stdio.h, which stands for "standard input/output header", is the header in the C standard library that contains macro definitions, constants, and declarations of functions and types used for various standard input and output operations.

Learn more problems on : Library Functions

Discuss about this problem : Discuss in Forum

20. What will be the output of the program?

```
#include<stdio.h>

int main()
{
    int i;
    char c;
    for(i=1; i<=5; i++)
    {
        scanf("%c", &c); /* given input is 'a' */
        printf("%c", c);
        ungetc(c, stdin);
    }
    return 0;
}
```

☐ A.aaaa ✗
☐ B.aaaaa ✓
☐ C.Garbage value. ✗
☑ D.Error in *ungetc* statement. ✗

Your Answer: Option D

Correct Answer: Option B

Explanation:

for(i=1; i<=5; i++) Here the for loop runs 5 times.

Loop 1:
scanf("%c", &c); Here we give 'a' as input.
printf("%c", c); prints the character 'a' which is given in the previous "scanf()" statement.
ungetc(c, stdin); "ungetc()" function pushes character 'a' back into input stream.

Loop 2:
Here the *scanf("%c", &c);* get the input from "stdin" because of "ungetc" function.
printf("%c", c); Now variable *c* = 'a'. So it prints the character 'a'.
ungetc(c, stdin); "ungetc()" function pushes character 'a' back into input stream.

This above process will be repeated in **Loop 3**, **Loop 4**, **Loop 5**.

Learn more problems on : Library Functions

Discuss about this problem : Discuss in Forum

***** END OF THE TEST *****

Feedback

Quality of the Test	:	-- Select -- ▼
Difficulty of the Test	:	-- Select -- ▼

Comments:

. . .

Submit Feedback

Current Affairs 2018

Interview Questions and Answers

© 2009 - 2018 by IndiaBIX™ Technologies. All Rights Reserved. | Copyright | Terms of Use & Privacy Policy

Contact us: info-@-@indiab@ix.com Follow us on twitter!