**ZOHO Questions – Day 1**

1. Find the output for the following programs(branching and looping)

#include<stdio.h>

Void main()

{

int i;

for(1=1;1<4;1++)

{

switch(1)

{

case 1: printf("%d", i);break;

case 2: printf("%d", i);break;

case 3: printf("%d", 1);break;

switch(1)

case 4: printf("%d", 1);break;

**Output: 1234**

2. Find the output(operartor and expression)

void main()

{

char s = "\12345s\n";

printf("%d", sizeof(s));

}

**Output: 4**

3. Find the output( Funtions)

int main()

{

static int 1 = 3;

printf("%d", i--);

return 1>0? main(): 0;

}

**Output: 321**

4. Find the output(pointers)

int main()

{

Millions of documents at your ingertips, ad-free.

Subscribe wit

char \*s[]={ "dharmr'a","hewlett-packard","siemens","ibm"};

char \*\*p;

p=s;

printf("%s",++\*p);

printf("%s","p++); ;

printf("%s",++\*p);

}

**Output: harmr'aharmr'aewlett-packard**

5. Find the output( dynamic memory)

#include<stdio.h>

#include<malloc.h>

#include<string.h>

int main()

{

int i;

char a[]="String";

char \*p = "New String";

char \*temp;

temp = malloc(strlen(p) + 1);

p = malloc(strlen(temp) + 1);

strcpy(p, temp);

printf("%s", p);

}

**Output: unpredictable string**

6. Find the output(algorithm)

int main()

{

int n = 12, res = 1;

while(n > 3)

{

n -= 3;

res \*= 3;

}

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

}

**Output: 81**

7. Find the output(function)

void fun(int [][3]);

int main()

{

int a[3][3] [9,8,7,6,5,4,3,2,1);

fun(a);

printf("%d\n", a[2][1]);

void fun(int b[][3])

{

++b;

b[1][1]=5;

}

**Output: 5**

8. Find the output(strings)

void main()

int i, n;

char x[5]:

strcpy(x, "Zoho");

n-strlen(x);

x(x+(n-1));

printf("%s", x);

**Output: ooho**

9. Find the output(arrays)

void main()

int c[] (5,4,3,4,5);

int j. \*q-c

for(j=0;j<5;j++){

printf("%d", \*c);

++q;

**Output:55555**

10. Find the output(branching and looping)

void main()

int i-1:

for(i=0; i--1;-1){

printf("%d", i);

if(i!-1) break;

1

**Output: -1**

11. Find the output(branching and looping)

void main()

int i--1:

printf("%d +1-%d\n",1,+1);

1

**Output: 1-1 1-1**

12. Find the output(datatypes)

void main()

{

char not;

not-12:

printf("%d".not);

1

**Output: 12**

13. Find the output(branching and looping)

#define FALSE -1

#define TRUE I

#define NULLO

void main()

if(NULL)

puts("NULL");

else if(FALSE)

puts("TRUE");

else

puts("FALSE");

1

**Output: TRUE**

14. Find the output(operator and expressions)

void main()

int k-1;

printf("%d-1 is""%s" k, k-1? "TRUE": "FALSE");

**Output: 1-1 is TRUE**

16. Find the output(branching and looping)

int main()

intt, 1;

for (t-4;scanf("%d",&i)-t;printf("%d\n",i))

printf("%d--".t-);

**Output: loop runs 4 timess**

16. Find the output(structures and unions)

struct emp

int len;

char name[1]:

int main()

1

char newname[] - "Rahul";

struct emp "p(struct emp) malloc(sizeof(struct emp) -1 + strlen(newname)+

1);

p->len strlen(newname);

strcpy(p->name, newname);

printf("%d %s\n", p->len, p->name); return 0;

1

**Output: 5 Rahul**

17. Find the output(datatypes)

void main()

{

float a;

a-4/2;

printf("%f%f\n",a.4/2);

}

**Output: 2.000000 0.000000**

18. Find the output(ooperator and expression)

void main()

{

printf("%d\n",sizeof(4)/sizeof(2.0));

printf("%d\n",sizeof(2.0)/sizeof(4));

}

**Output: 02**

19. Output of this program?

```cpp

#include <iostream>

using namespace std;

int main()

{

int i, j, var = 'A';

for (i = 5; i >= 1; i--) {

for (j = 0; j < i; j++)

printf("%c ", (var + j));

printf("\n");

}

return 0;

}

```

**Output:**

**A B C D E**

**A B C D**

**A B C**

**A B**

**A**

20. Output of the program?

```c

#include <stdio.h>

void f(char\*\*);

int main()

{

char \*argv[] = { "ab", "cd", "ef", "gh", "ij", "kl" };

f(argv);

return 0;

}

void f(char \*\*p)

{

char \*t;

t = (p += sizeof(int))[-1];

printf("%s\n", t);

}

```

**Output:**

**gh**

21. Output of the following program?

```c

#include <stdio.h>

void dynamic(int s, ...)

{

printf("%d ", s);

}

int main()

{

dynamic(2, 4, 6, 8);

dynamic(3, 6, 9);

return 0;

}

```

**Output:**

**2 3**

22. Predict the output?

```c

#include <stdio.h>

int main()

{

void demo();

void (\*fun)();

fun = demo;

(\*fun)();

fun();

return 0;

}

void demo()

{

printf("program ");

}

```

**Output:**

**program program**

23. Explain: What is the output of the following program?

```c

#include <stdio.h>

int main()

{

char str1[] = "ZohoInterview";

char str2[] = {'t', 'e', 's', 't', 't', 'e', 's', 't', '1'};

int n1 = sizeof(str1)/sizeof(str1[0]);

int n2 = sizeof(str2)/sizeof(str2[0]);

printf("n1 = %d, n2 = %d", n1, n2);

return 0;

}

```

**Output:**

**n1 = 14, n2 = 9**

Explanation:

- `str1` is a null-terminated string, so its length is 14 (13 characters + 1 null terminator).

- `str2` is an array of characters without a null terminator, so its length is 9.

24. Question 1

```cpp

#include <iostream>

using namespace std;

int a = 20;

int main() {

int a = 10;

std::cout << a << "" << ::a;

return 0;

}

```

**Output:**

**1020**

25. Question 2

```cpp

#include<iostream>

void func(int \*b){

\*b = 1;

}

int main() {

int \*a;

int n;

a = &n;

\*a = 0;

func(a);

std::cout << \*a << std::endl;

return 0;

}

```

**Output:**

**1**

26. Question 3

```cpp

#include <iostream>

using namespace std;

class Test {

static int i;

int j;

};

int Test::i;

int main() {

cout << sizeof(Test);

return 0;

}

```

**Output**

**4**

27. Question 4

```c

#include <stdio.h>

int main() {

int i;

char ch[] = {'z','o','h','o'};

char \*ptr, \*str1;

ptr = ch;

str1 = ch;

i = (\*ptr-- + ++\*str1) - 10;

printf("%d", i);

return 0;

}

```

**Output**

**235**

28. Question 5

```c

int main() {

int a[10][10] = {{1,2},{3,4},{5,6},{7,8},{9,10}};

int \*p = a[3];

int result = (\*p + 2) \* a[4][1] + (++\*p) + (\*p + 7);

printf("%d", result);

return 0;

}

```

**Output**

**113**

29

```c

switch(5)

{

default:printf("five");

case 1:printf("one");

}

```

**Output:**

**`fiveone`**

Explanation: The `default` case gets executed since there's no matching case for `5`. After `default`, execution falls through to `case 1`, resulting in printing `fiveone`.

30

```c

if(101)

printf("hai");

else

printf("all");

```

**Output:**

**`hai`**

Explanation: `101` is a non-zero value, which is considered `true` in C. Therefore, `printf("hai");` is executed.

31

```c

for(i=0;i<5;i++);

printf("%d",i);

```

**Output: `5`**

Explanation: The semicolon after the `for` loop creates an empty loop body. After the loop, `i` is `5`, which gets printed.

32

```c

void main()

{

int a=10,b=20,c=30;

printf("%d%d");

}

```

**Output:**

**Undefined behavior.**

Explanation: The `printf` function is missing the variables to be printed, leading to undefined behavior.

33

```c

main()

{

int i=0;

switch(i)

{

case 0: i++;

printf("%d", i);

case 1:

printf("%d", i);

case 2:

printf("%d", i);

}

}

```

**Output:**

**`111`**

Explanation: `case 0` increments `i` to `1`, then falls through to `case 1` and `case 2`, printing `1`, `1`, `2` in sequence.

34

```c

void main()

{

printf("\n98");

printf("\b76");

printf("\r54");

}

```

**Output: `\r54` (results depend on terminal).**

Explanation: `\b` moves the cursor back, `\r` moves to the start of the line.

35

```c

int main()

{

int x=3;

if(x==2); x=0;

if(x==3) x++;

else

printf("x=%d",x);

}

```

**Output: `1`**

Explanation: First `if` does nothing (semicolon); second `if` is false; `else` not executed.

36

```c

int main()

{

printf(" \" hai %% all \" ");

}

```

**Output: `" hai % all "`**

Explanation: Escape sequences print double quotes and percent sign.