1.
```c
#include <stdio.h>
int main() {
  int a = 5;
  if (a++ == 5 && ++a == 7)
      printf("True: a = %d\n", a);
  else
      printf("False: a = %d\n", a);
  return 0;
    }
```

2.
```c
#include <stdio.h>

void update(int *p) {
  int x = 99;
  p = &x;
  *p = 500;
}

int main() {
  int a = 10;
  update(&a);
  printf("%d", a);  // Output?
}
```

3.
```c
#include <stdio.h>
int main() {
  int x = 0;
  while (x = x + 1) {
      if (x == 3) break;
  }
  printf("x = %d\n", x);
  return 0;
  }
```

4.
```c
#include <stdio.h>
int main() {
  char c = 255; // signed char wraps to -1 on most systems
  if (c > 0)
      printf("Positive\n");
  else
      printf("Negative or Zero\n");
  return 0;
 }
```

5.
```c
#include <stdio.h>
int main() {
  int x = 5, y = 10, z;
  z = (x > y) ? x : y, x + y;
  printf("z = %d\n", z);
  return 0;}
```

6.
```
#include<iostream>
    int main() {
  double a = 0.1;
  double sum = 0.0;
  for (int i = 0; i < 10; i++) {
      sum += a;
  }
  if (sum == 1.0) {
      std::cout << "Sum is 1.0";
  } else {
      std::cout << "Sum is not 1.0";
  }
  return 0;
 }
```

7.
```
#include <stdio.h>

 void foo() {
   printf("foo ");
   return;
   printf("after return ");
  }

   int main() {
   foo();
  }
```

8.
```
#include <stdio.h>

int fun(int x) {
  if (x == 0)
      return 0;
  return x + fun(x--);
}

int main() {
  printf("%d", fun(5));
}
```

9.
```
#include <stdio.h>

int x = 5;

void test() {
  int x = 10;
```

```
    printf("%d ", x);
  }

  int main() {
    test();
    printf("%d", x);
  }
```

10.
```
#include <stdio.h>

 int fun() {
  static int count = 0;
  return ++count;
 }

 int main() {
  printf("%d ", fun());
  printf("%d ", fun());
  printf("%d", fun());
}
```

11.
```
#include <stdio.h>

 void check(int arr[]) {
  printf("In function, sizeof(arr) = %lu\n", sizeof(arr));
 }

 int main() {
  int arr[] = {1, 2, 3, 4};
  printf("In main, sizeof(arr) = %lu\n", sizeof(arr));
  check(arr);
}
```

12.
```
#include <stdio.h>

 int main() {
  int arr[] = {1, 2, 3};
  int *ptr = arr;

  printf("sizeof(arr) = %lu\n", sizeof(arr));
  printf("sizeof(ptr) = %lu\n", sizeof(ptr));
 }
```

13.
```
#include <stdio.h>

 int add(int a, int b) {
   return a + b;
}

int main() {
   int (*fptr)(int, int) = add;
   printf("%d", fptr(2, 3));}
```

14.
```
#include <stdio.h>

int sum(int a, int b) {
 return a + b;
}

int main() {
  printf("%d", sum(5, sum(2, 3)));
}
```

15.
```
#include <stdio.h>

int main() {
  char str[] = "hello";
  char *p = str;

  printf("%c %c\n", *str, *p);
}
```

16.
```
#include <stdio.h>
#include <string.h>

int main(void) {
    char buf[10] = {1, 2, 3, 4, 5, 6, 9, 8};
                                                    // Only 8
initialized, rest are 0
    char p = *(buf + 1 + 5);
                                                    // Access 6th
element after buf + 1
    printf("%d\n", p);
    return 0;
}
```

17.
```
#include<stdio.h>
 #include<stdlib.h>
void main()
{
int n;
 printf("Enter a number:");
 scanf("%d",&n);
printf("%d", (n<<3)+n);
  getch();
}
```

18.
```
#include <stdio.h>

int mystery(int n) {
 if (n == 0) return 0;
 return n + mystery(n - 1);
}

int main() {
```

```
    printf("%d", mystery(4));
  }
```

19.
```
static int x = 10;
void func() {
 static int x = 20;
 printf("%d ", x);
}
int main() {
 func();
 printf("%d", x);
}
```

20.
```
#include <stdio.h>

int x = 0;

void recur() {
 if (x == 3) return;
 x++;
 recur();
 printf("%d ", x);
}

int main() {
 recur();
}
```

21.
```
#include <stdio.h>

 int main() {
 int i, k;
 for (i = 0, k = 0; (i < 5 && k < 3); k++, i++) {
     printf("%d ", i);
 }
 return 0;
}
```

22.
```
void recursive() {
 static int num = 3;
 if (num--) {
     printf("%d ", num);
     recursive();
 }
}
 int main() {
 recursive();
}
```

23.
```
void func() {
 static int i = 5;
 i++;
```

```
    printf("%d ", i);
    }
     int main() {
    func(); func(); func();
    }
```

24.
```
#include <stdio.h>

   void fun(int **p) {
    static int q = 10;
    *p = &q;
                       // Now the pointer in main points to q
   }

    int main() {
    int r = 20;
    int *p = &r;
        // p points to r
    fun(&p);
                                       // p is redirected to point to q
    printf("%d %d", *p, r);
    return 0;
   }
```

25.
```
#include <stdio.h>

   void updateArray(int arr[]) {
    arr[0] = 99;

                  // affects original
   }

    void updatePointer(int *ptr) {
    ptr = ptr + 1;
                                                                        //
only modifies local copy of pointer
    *ptr = 88;

      // doesn't affect arr[0]
   }

   int main() {
    int arr[] = {1, 2, 3};
    updateArray(arr);
    updatePointer(arr);
    printf("arr[0] = %d\n", arr[0]);
                                                                  // ?
   }
```

26.
```
#include <stdio.h>
   int  main() {
    int i = 0;
    for (; i++ < 5; printf("%d ", i));
```

```
    return 0;
  }
```

27.
```
#include <stdio.h>

 void swap(int a, int b) {
  int temp = a;
  a = b;
  b = temp;
 }

  int main() {
  int x = 5, y = 10;
  swap(x, y);
  printf("%d %d", x, y);
 }
```

28.
```
#include <stdio.h>
 #include <string.h>

 static int fn(int x, int y) {
 return x < y;
                                          // returns 1 if true, 0 if
false
 }

 int main() {
  int n = 0, a = 3, b = 8;
  while (n <= fn(b, a)) {
      n++;
      printf("%d", n);
  }
  return 0;
 }
```

29.
```
#include <stdio.h>

 void change(int *a, int b) {
  *a = 10;
  b = 10;
}

 int main() {
  int a = 100;
  int b = 50;
  change(&a, b);
  printf("%d %d", a, b);
  return 0;
}
```

30.
```
void demo() {
 auto int a = 1;
```

```
 static int b = 1;
 a++;
 b++;
  printf("%d %d\n", a, b);
 }
int main() {
 demo(); demo();
}
```

31.
```
void demo() {
 static int count = 0;
 count++;
 printf("%d ", count);
 }
 int main() {
 demo(); demo(); demo();
}
```

32.
```
#include <iostream>

using namespace std;

void modify(int &x) {

  x = x * 2;

  }

  int main() {

  int a = 5;

   modify(a);

   cout << a << endl;

   return 0;

  }
```

33.
```
#include <stdio.h>

int main() {
int x = 42;
void *ptr = &x; // generic pointer

printf("Address of x: %p\n", ptr);

// Cast to int* to dereference
printf("Value of x: %d\n", *(int *)ptr);

return 0;
}
```

34.
```
#include<iostream>
```

```
int main() {
 int a = -5;
 unsigned int b = 10;
 std::cout << (a + b);
 return 0;
}
```

35.
```
#include <stdio.h>

 void change(int *a, int *b) {
 *a = 10;
 *b = 20;
  }

 int main() {
 int x = 1, y = 2;
 change(&x, &y);
 printf("%d %d", x, y); // Output?
}
```

36.
```
#include <stdio.h>

 int print() {
 printf("print ");
 return 1;
 }

 int main() {
 if (print() && print())
     printf("main");
 }
```

37.
```
#include<iostream>
int main() {
 short a = 32767;
 int b = a + 1;
 std::cout << b;
 return 0;
 }
```

38.
```
#include <stdio.h>

int main() {
 void test() {
```

```
    printf("Hello World");
}
test();
}
```

39.
```
#include <stdio.h>

int func(int i) {
 if (i % 2) {
     return i++;
 } else {
     return func(func(i - 1));
 }
 }

int main() {
 printf("%d %d", func(200), func(201));
 return 0;
}
```

40.
```
#include<iostream>

   int main() {

bool b = -1;

if(b){ std::cout << "true\n" ;} else {std::cout << "false\n";}

std::cout << b;

return 0;

}
```

41.
```
#include <stdio.h>
int main() {
static int array[] = {10, 20, 30, 40, 50};
printf("%d %d", *array, *(array + 3) * *array);
return 0;
}
```

42.
```
#include <stdio.h>
#define fn(x) ((x) & (x - 1))

int main() {
printf("%d", fn(12) * fn(14) * fn(16));
return 0;
}
```

43.
```
#include <stdio.h>

void update(int arr[]) {
arr[0] = 99;
```

```
}

 int main() {
   int arr[] = {1, 2, 3};
   update(arr);
   printf("%d", arr[0]);  // Output?
}
```

44.
```
#include <stdio.h>

static int i = 25;
          // Global static variable

void func() {
 printf("%d\n", i);
              // Prints global 'i'
 return;
 }

int main() {
 static int i;
                     // Local static variable, initialized to 0
 printf("%d ", i);
                     // Prints local 'i'
 func();
              // Calls func() which prints global 'i'
 return 0;
}
```

45.
```
#include <stdio.h>

 void modify(int **p) {
 static int x = 200;
 *p = &x;
 }

 int main() {
 int a = 10;
 int *ptr = &a;
 modify(&ptr);
 printf("%d", *ptr);  // Output?
}
```

46.
```
#include <stdio.h>

int fun() {
 printf("hello");
 }

int main() {
 int x = fun();
 printf("%d", x);}
```

47.
```
void func() {
 static int x;
 printf("%d ", x);
}
int main() {
 func();
}
```

48.
```
void demo() {
 auto int a = 0;
 a++;
 printf("%d ", a);
}
int main() {
 demo(); demo();
}
```