

CS 137

Making Decisions

Fall 2025

Victoria Sakhnini

Table of Contents

if statement.....	2
if ... else statement.....	3
if ... else if ... else statement.....	4
Nested if statements.....	5
Ternary Conditional Operator.....	6
Switch statement	7
Additional Examples.....	10
Extra Practice Problems	11

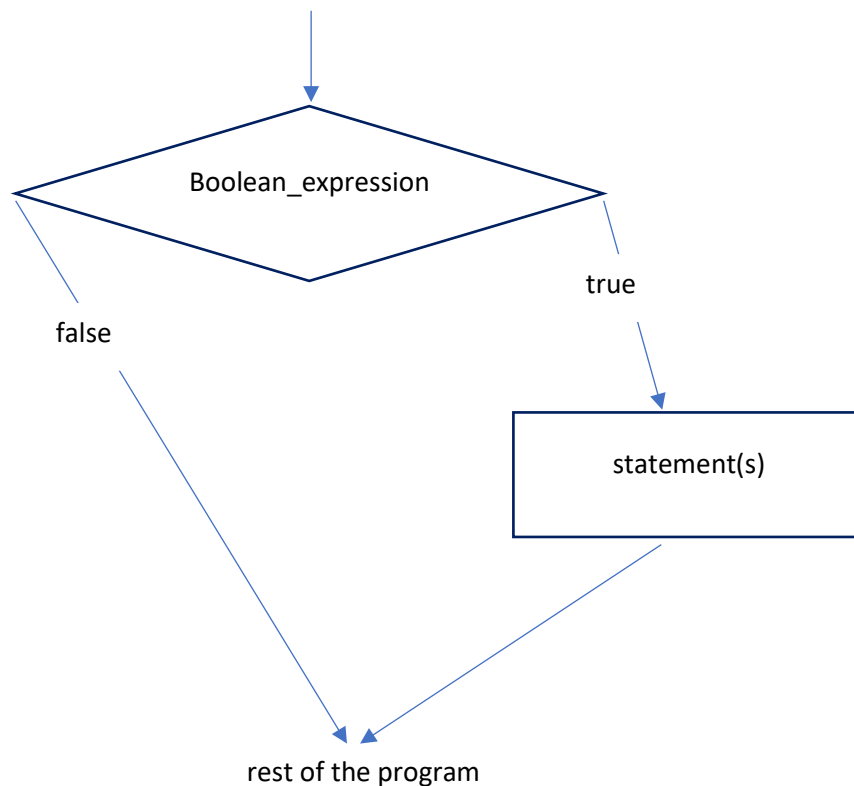
if statement

An `if` statement consists of a boolean expression (its true or false value) followed by one or more statements.

Syntax:

```
if(boolean_expression) {  
/* statement(s) will execute if the boolean expression is true */  
}
```

If the boolean expression evaluates to `true`, then the code block inside the `if` statement is executed. If the boolean expression evaluates to `false`, then the code block inside the `if` statement is skipped(ignored). C programming language assumes any non-zero values as `true`; if it is zero, it is assumed as `false`.



Example:

```
1. #include <stdio.h>  
2. int main(void)  
3. {  
4.     printf("Enter an integer: ");  
5.     int x;  
6.     scanf("%d", &x);  
7.     if (x < 0)  
8.         x *= -1;  
9.     printf("x=%d\n", x);  
10. }
```

Console program output

```
Enter an integer: -10  
x=10  
Press any key to continue...
```

Console program output

```
Enter an integer: 7  
x=7  
Press any key to continue...
```

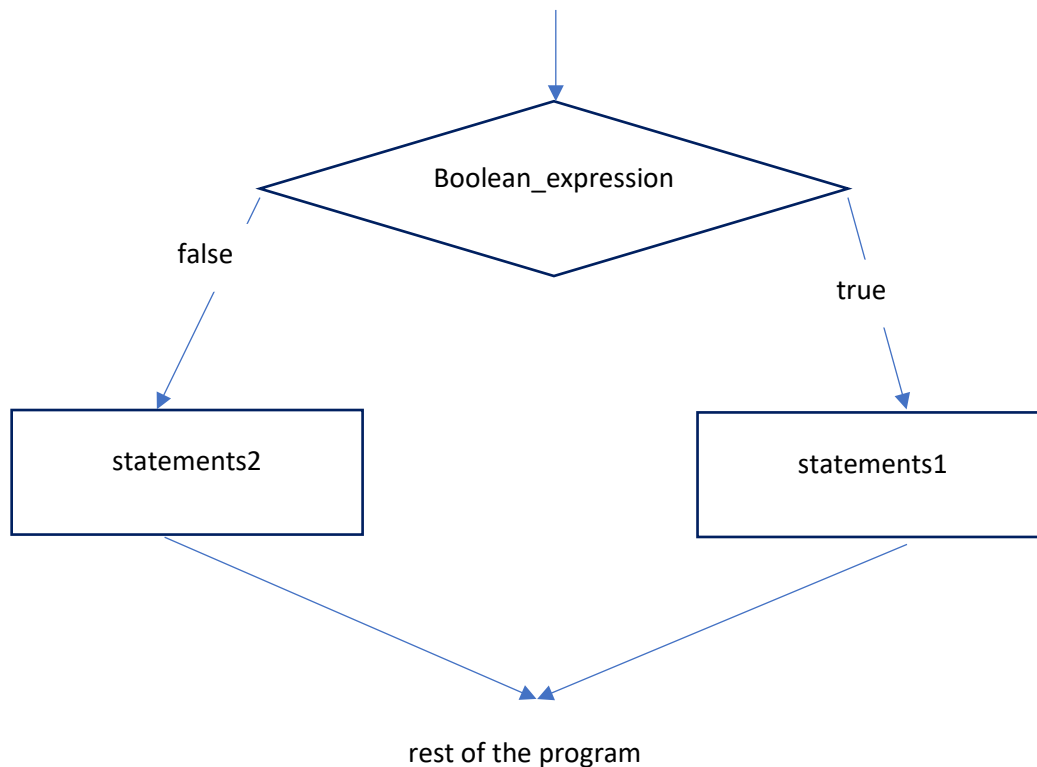
if ... else statement

An `if` statement can be followed by an optional `else` statement, which executes when the boolean expression is false.

Syntax:

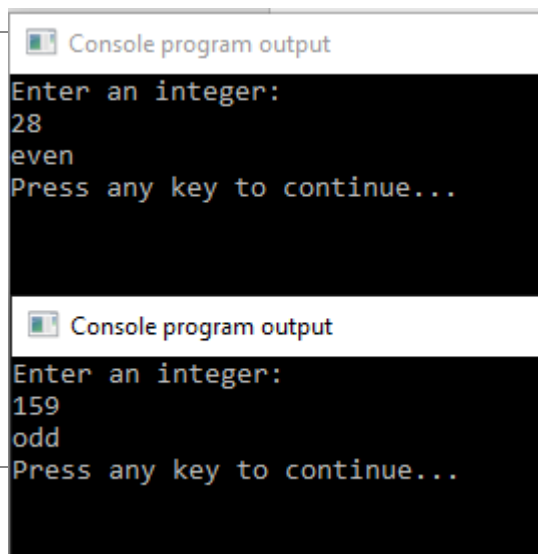
```
if (boolean_expression){
    statements1 /* will execute if the boolean expression is true */
}
else{
    statements2 /* will execute if the boolean expression is false */
}
```

If the boolean expression evaluates to `true`, then the `if` block of code is executed; otherwise, `else` block of code is executed.



The following example is of a program that reads an integer and prints if it is odd or even.

```
1. #include <stdio.h>
2. int main(void)
3. {
4.     int a;
5.     printf("Enter an integer: \n");
6.     scanf("%d", &a);
7.     if (a%2 == 0){
8.         printf("even\n");
9.     }
10.    else{
11.        printf("odd\n");
12.    }
13.    return 0;
14. }
```



if ... else if ... else statement

An if statement can be followed by an optional `else if...else` statement, which is very useful to test various conditions using single `if...else if` statement.

When using `if`, `else if`, `else` statements, there are a few points to keep in mind:

- An if can have zero or one else's, and it must come after any else if's.

- An if can have zero to many else if's, and they must come before the else.


- Once an else if succeeds, none of the remaining else if's or else's will be tested.

Syntax:

```
if (boolean_expression1) {
/* Executes when the boolean expression 1 is true */
} else if (boolean_expression2) {
/* Executes when the boolean expression 2 is true (and the above boolean expressions are false) */
} else if (boolean_expression3) {
/* Executes when the boolean expression 3 is true (and the above boolean expressions are false) */
} else {
/* executes when none of the above conditions is true */
}
```

The following example is a program that reads an integer representing a student's score and prints the corresponding letter grade using 90/80/70/60 scale.

```
1. #include <stdio.h>
2. int main(void)
3. {
4.     int score;
5.     char grade;
6.     printf("Enter a numerical grade between 0-100: \n");
7.     scanf("%d", &score);
8.     if (score >= 90)
9.         grade = 'A';
10.    else if (score >= 80)
11.        grade = 'B';
12.    else if (score >= 70)
13.        grade = 'C';
14.    else if (score >= 60)
15.        grade = 'D';
16.    else grade = 'F';
17.    printf("The letter grade is: %c\n", grade);
18.    return 0;
19. }
```

 `%c` is used to print a character.

Nested if statements

It is always legal in C programming to nest if-else statements, which means you can use if or else if statement inside another if or else if statement(s).



For each of the provided sample outputs, trace it manually and understand the results.

```
1. #include <stdio.h>
2. int main(void)
3. {
4.     int var1, var2;
5.     printf("Input the value of var1:");
6.     scanf("%d", &var1);
7.     printf("Input the value of var2:");
8.     scanf("%d", &var2);
9.     if (var1 != var2)
10.    {
11.        printf("var1 is not equal to var2\n");
12.        //Nested if else
13.        if (var1 > var2)
14.        {
15.            printf("var1 is greater than var2\n");
16.        }
17.        else
18.        {
19.            printf("var2 is greater than var1\n");
20.        }
21.    }
22.    else
23.    {
24.        printf("var1 is equal to var2\n");
25.    }
26.    return 0;
27. }
```

Console program output

```
Input the value of var1:876
Input the value of var2:-20
var1 is not equal to var2
var1 is greater than var2
Press any key to continue...
```

Console program output

```
Input the value of var1:23
Input the value of var2:87
var1 is not equal to var2
var2 is greater than var1
Press any key to continue...
```

Console program output

```
Input the value of var1:99
Input the value of var2:99
var1 is equal to var2
Press any key to continue...
```



What is the output of the following program¹?

```
1. #include <stdio.h>
2. int main(void)
3. {
4.     int i = 3;
5.     if (i % 2 == 0)
6.         if (i == 0)
7.             printf(" zero \n");
8.     else
9.         printf("how odd \n");
10.    return 0;
11. }
```

¹ This is the dangling 'else' issue. This code prints nothing. Each else statement, independent of indentation belongs to the nearest if statement. Code can be very confusing to read without brackets and you are advised to use them judiciously to help with clarity. In this example else belongs to the last if, this since `i%2==0` is false, the next statement to be executed is `return 0`

Ternary Conditional Operator


Syntax:

```
expr1 ? expr2 : expr3;
```

If `expr1` is true, this conditional statement returns the value of `expr2`. Otherwise, it returns `expr3`.

Example:

```
1. #include <stdio.h>
2. int main(void)
3. {
4.     int a, b, c, d, m1, m2;
5.     printf("Enter four integers: ");
6.     scanf("%d %d %d %d", &a, &b, &c, &d);
7.     m1 = a > b ? a : b;
8.     m2 = c > d ? c : d;
9.     printf("The maximal value: %d\n", m1 > m2 ? m1 : m2);
10.    return 0;
11. }
```

 Console program output

```
Enter four integers: 10 15 -6 0
The maximal value: 15
Press any key to continue...
```

Switch statement

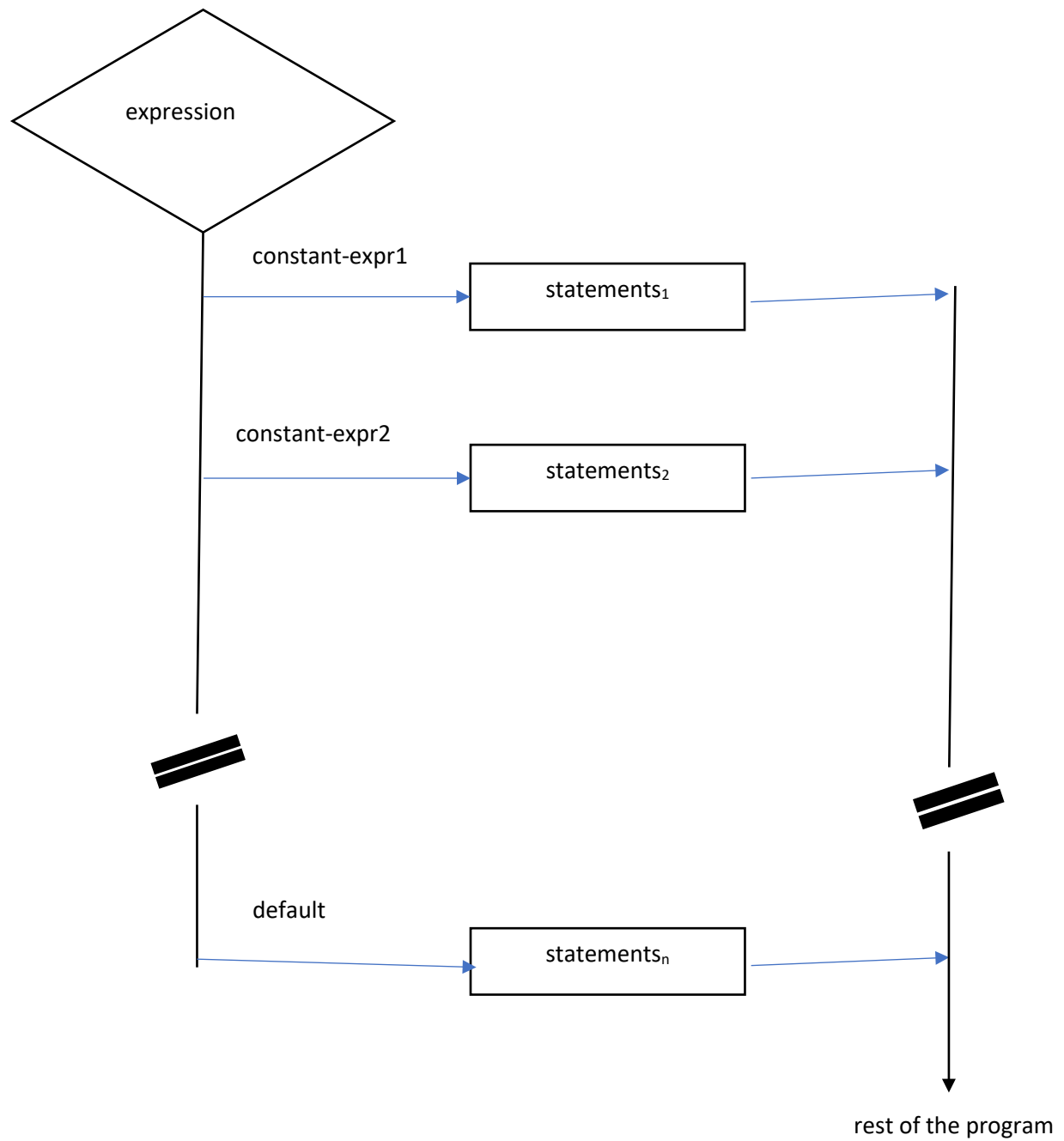
A `switch` statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each `switch` case.

Syntax:

```
switch(expression)
{
    case constant-expr1:
        statements1;
        break;           /* optional */
    case constant-expr2:
        statements2;
        break;           /* optional */
    /* you can have any number of case statements */
    default:              /* Optional */
        statementsn;
}
```

The following rules apply to a `switch` statement:

- You can have any number of case statements within a `switch`.
- The constant-expression for a case must be the same data type as the variable in the `switch`.
- When the variable being switched on is equal to a case, the statements following that case will execute until a `break` statement is reached.
- When a `break` statement is reached, the `switch` terminates, and the control flow jumps to the following line following the `switch` statement.
- Not every case needs to contain a `break`. If no `break` appears, the control flow will fall through to the following cases until a `break` is reached.
- `default` case is optional, which must appear at the end of the `switch`. The default case can be used for performing a task when none of the cases is true. No `break` is needed in the `default` case.



Example: (next page)

```

1. #include <stdio.h>
2. int main(void)
3. {
4.     // local variable definition
5.     char grade;
6.     printf("Enter a grade.\n");
7.     scanf("%c", &grade);
8.     switch (grade)
9.     {
10.         case 'A':
11.             printf("Excellent!\n");
12.             break;
13.         case 'B':
14.         case 'C':
15.             printf("Well done\n");
16.             break;
17.         case 'D':
18.             printf("You passed\n");
19.             break;
20.         case 'F':
21.             printf("Better try again\n");
22.             break;
23.         default:
24.             printf("Invalid grade\n");
25.     }
26.     printf("Thank you!\n");
27.     return 0;
28. }

```

Console program output

```

Enter a grade.
K
Invalid grade
Thank you!
Press any key to continue...

```

Console program output

```

Enter a grade.
B
Well done
Thank you!
Press any key to continue...

```

Select Console program output

```

Enter a grade.
a
Invalid grade
Thank you!
Press any key to continue...

```

Select Console program output

```

Enter a grade.
D
You passed
Thank you!
Press any key to continue...

```



What happened when the input was a ²?



What happened when the input was B ³?

² None of the cases is true thus default case was executed.

³ The second case was true but there is no break there so the following case was executed as well until break statement is reached.

Additional Examples

```
/*
 * Transforms a compass heading to a compass bearing using this table:
 *
 * HEADING
 * IN DEGREES      BEARING COMPUTATION
 *
 * 0   - 89.999... north (heading) east
 * 90  - 179.999... south (180.0 - heading) east
 * 180 - 269.999... south (heading - 180.0) west
 * 270 - 360       north (360.0 - heading) west
 */
#include <stdio.h>
void instruct(void);

int main(void)
{
    double heading; /* Input - compass heading in degrees */

    /* Get compass heading. */
    printf("Enter a compass heading> ");
    scanf("%lf", &heading);

    /* Display equivalent compass bearing. */
    if (heading < 0.0)
        printf("Error -- negative heading (%.1f)\n", heading);
    else if (heading < 90.0)
        printf("The bearing is north %.1f degrees east\n", heading);
    else if (heading < 180.0)
        printf("The bearing is south %.1f degrees east\n", 180.0 - heading);
    else if (heading < 270.0)
        printf("The bearing is south %.1f degrees west\n", heading - 180.0);
    else if (heading <= 360.0)
        printf("The bearing is north %.1f degrees west\n", 360.0 - heading);
    else
        printf("Error--heading > 360 (%.1f)\n", heading);
    return (0);
}
```

Console program output

```
Enter a compass heading> 110
The bearing is south 70.0 degrees east
Press any key to continue...
```

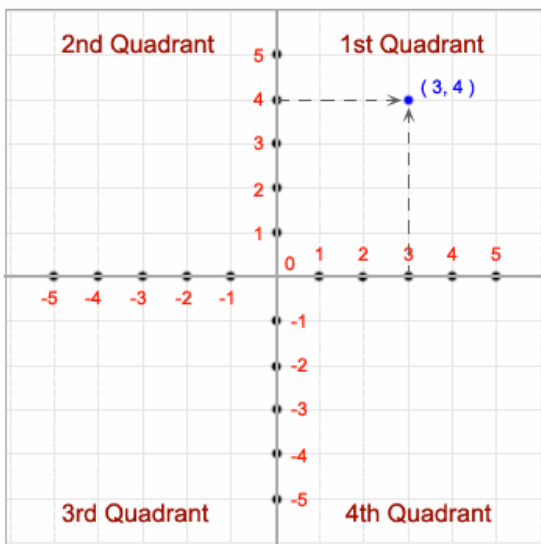
Extra Practice Problems

1) Write a program that reads three positive integers (assume they are different) and prints the middle value. For example, if the program reads 10 30 12, it prints 12 because $10 < 12 < 30$.ⁱ

2) a) If you solved Q1 using nested if... solve it again without nested if.ⁱⁱ

b) How many unique tests do you need for your programⁱⁱⁱ

3) a) Write a program that reads two integers representing a coordinate point(x,y) and prints in which quadrant the (x,y) point lies or whether it lies on the x-axis or y-axis. For example, if the input is 3 4, the output should be 1st quadrant



b) How many test cases should you use at least to test your solution thoroughly⁴?

⁴ At least 6 test cases. One for each quadrant and one for each axis.

4) Write a program that reads an integer temperature in Celsius and prints one of the following messages:

Freezing, if temperature < 0

Very cold, if the temperature is between 1-10

Cold, if the temperature is between 11-20

Nice, if the temperature is between 21-29

Hot, if the temperature is between 30-40

Very hot if the temperature is at least 40.

5) Write a program `add_check.c` that reads a 4-digit natural number `n` and adds a check digit to it if `n` is invalid (to the right, i.e., the “ones” column). If `n` is valid, the number is going to remain the same. Your program should print the final value with a newline character.

Sample input 1: 8675

Sample output 1: 86754

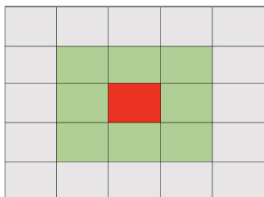
Explanation: Notice that $8 + 6 + 7 + 5 = 26$. This is 4 away from 30, so the check digit must be 4

Sample input 2: 5555

Sample output 2: 5555

Explanation: Notice that $5 + 5 + 5 + 5 = 20$. This is divisible by 10, so the number is considered to be valid.

6) Yuhan and her two friends were getting ready to go to her favorite artist’s mini-meet. The arena is a 5x5 square area, with the red center square being the stage.



The grey-shaded area represents the edge seats, and the green area represents the inner seats. The grey area costs \$40, and the green area costs \$60. Create a function `ticket` that takes in the coordinates of the seats of all three friends one by one and prints the total cost of the mini-meet.

Example input:

0 0

1 1

2 1

Output: 160

So here the coordinates are (0,0)(1,1)(2,1).

7) Write a program 'cinema.c' that reads two integers, 'age' and 'time', representing a person's age and the time (in 24-hour format) they want to watch a movie. The program should print the ticket price based on the following rules:

- * If the person is under 13 or over 60, the ticket is \$5.

- * For everyone else, the ticket is \$10.

- * However, if the movie is scheduled between 8 PM (20:00) and midnight (24:00), there's a \$2 discount on the ticket.

#Sample Input #1:

12 15

#Sample Output #1:

5

#Sample Input #2:

30 21

#Sample Output #2:

8

#Sample Input #3:

65 19

#Sample Output #3:

5

i

```
#include <stdio.h>
int main(void)
{
    int a, b, c;
    printf("Enter three different integers: ");
    scanf("%d %d %d", &a, &b, &c);
    if (a > b)
        if (a < c)
            printf("The middle value is %d\n", a);
        else if (b > c)
            printf("The middle value is %d\n", b);
        else
            printf("The middle value is %d\n", c);
    else if (a > c)
        printf("The middle value is %d\n", a);
    else if (b < c)
        printf("The middle value is %d\n", b);
    else
        printf("The middle value is %d\n", c);
    return (0);
}
```

ii

```
#include <stdio.h>
int main(void)
{
    int a, b, c;
    printf("Enter three different integers: ");
    scanf("%d %d %d", &a, &b, &c);
    if (a > b && a < c)
    {
        printf("The middle value is %d\n", a);
        return (0);
    }
    if (a > b && b > c)
    {
        printf("The middle value is %d\n", b);
        return (0);
    }
    if (a > b && b < c)
    {
        printf("The middle value is %d\n", c);
        return (0);
    }
    if (a < b && a > c)
    {
        printf("The middle value is %d\n", a);
        return (0);
    }
    if (a < b && b < c)
    {
        printf("The middle value is %d\n", b);
        return (0);
    }
    if (a < b && b > c)
    {
        printf("The middle value is %d\n", c);
        return (0);
    }
}
```

iii

6 tests