1.10 Intrinsic (built-in) Functions

The table below shows some of the intrinsic functions in FORTRAN 90 :

Function	Description	Example
sin(x)	sine of x (x in radians)	sin(2) returns 0.909297
cos(x)	cosine of x (x in radians)	cos (2) returns -0.416147
tan(x)	tangent of x (x in radians)	tan(2) returns -2.18504
asin(x)	inverse sine of x (x in radians)	asin(0.2) returns 0.201358
acos(x)	inverse cosine of x (x in radians)	acos(0.2) returns 1.36944
atan(x)	inverse tangent of x (x in radians)	atan(0.2) returns 0.197396
exp(x)	exponential of x (base e)	exp(2) returns 7.38906
abs(x)	absolute value of x	abs(-2) returns 2
log(x)	natural logarithm of x (base e)[Ln(x)]	log(2) returns 0.693147
log10(x)	common logarithm of x (base 10)	Logl0(2) returns 0.30103
sqrt(x)	square root of x	sqrt(2) returns 1.41421
int(x)	Truncate to an integer	int(2.53) returns 2
real(x)	Convert an integer to real value	real(4) returns 4.0
nint(x)	Nearest integer of a real value	nint(2.53)=3
mod(x,y)	This operator is called the remainder or the modulus operator. It is used to find the remainder after the division. This operator cannot be used with real variables.	Remainder = mod(a , b)
real(C)	The real part of the complex number C	x= *real(2,-4)=2*
imag(C)	The imaginary part of the complex number C	y= *imag(2,-4)* = -4
conjg(C)	The complex conjugate of the complex number C	cc= *conjg(C)* =(2,4)

H.W Write a program to find the value of C form the following formula.

$$\mathrm{C} = \frac{4 + \frac{17}{M}}{\sqrt{N}} \quad , \qquad M = 4 \quad , N = 5$$

H.W Write a program that reads a temperature in Fahrenheit degrees and convert it into Celsius degrees, using the formula

$$C^0 = \frac{9}{5}(F^0 - 32)$$

H.W: 6Ω, 3Ω resisters are connected in series across a 36v source, write a program to find the total current and the voltage of each resister.

H.W: What are the values of the following logical expressions?

15>23

(12+3) <=15

(2>1) .AND. (3<4)

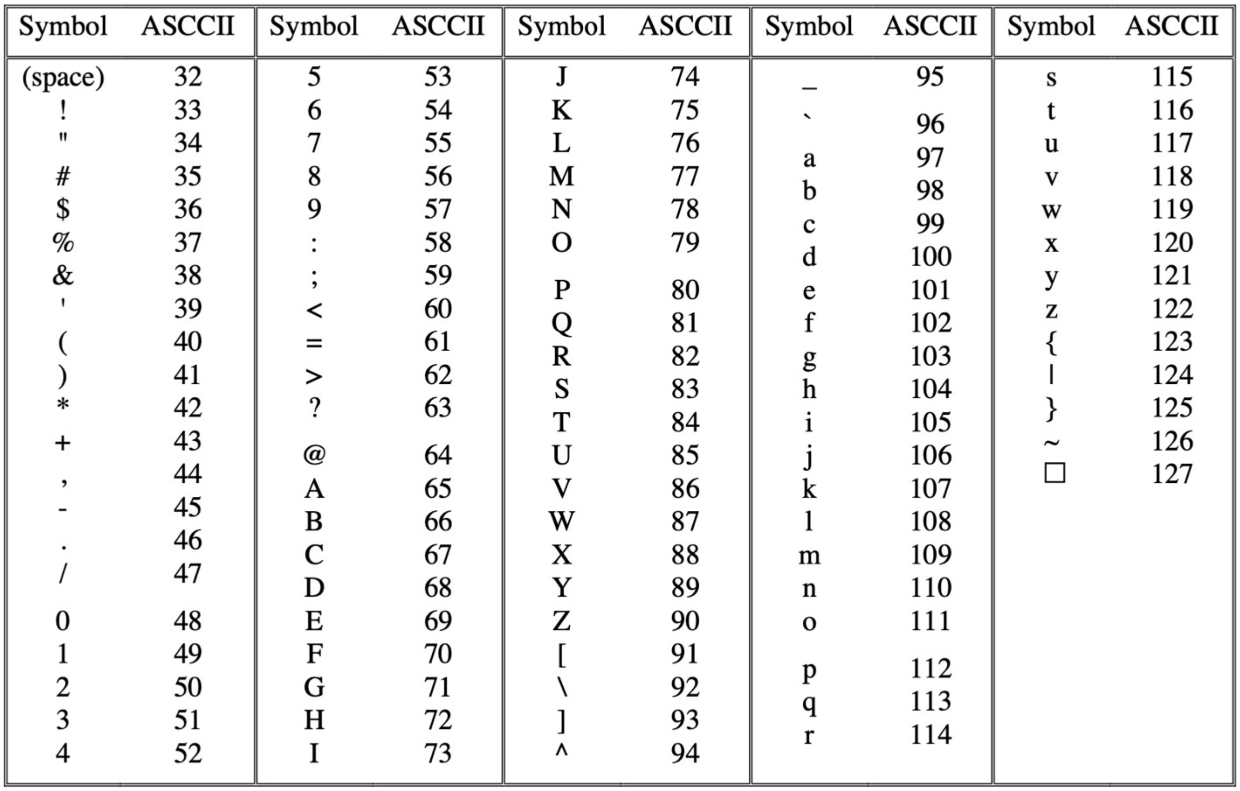
(3>2) .AND. (1+2)<3 .OR. (4<=3)

"Adam" > "Eve"

"ADAM" > "Adam"

"M1" < "M25"

Table of Symbols and the Corresponding ASCCII Codes



Symbol	ASCCII	Symbol	ASCCII	Symbol	ASCCII	Symbol	ASCCII	Symbol	ASCCII
(space)	32	5	53	J	74	_	95	s	115
!	33	6	54	K	75	`	96	t	116
"	34	7	55	L	76	a	97	u	117
#	35	8	56	M	77	b	98	v	118
$	36	9	57	N	78	c	99	w	119
%	37	:	58	O	79	d	100	x	120
&	38	;	59	P	80	e	101	y	121
'	39	<	60	Q	81	f	102	z	122
(	40	=	61	R	82	g	103	{	123
)	41	>	62	S	83	h	104	|	124
*	42	?	63	T	84	i	105	}	125
+	43	@	64	U	85	j	106	~	126
,	44	A	65	V	86	k	107	□	127
-	45	B	66	W	87	l	108		
.	46	C	67	X	88	m	109		
/	47	D	68	Y	89	n	110		
0	48	E	69	Z	90	o	111		
1	49	F	70	[	91	p	112		
2	50	G	71	\	92	q	113		
3	51	H	72	]	93	r	114		
4	52	I	73	^	94				

2. Conditional (Selection) Statements:

Another important technique when writing FORTRAN 90 program is the ability to select different paths of execution. FORTRAN 90 provides four selection constructs (statements), they are:

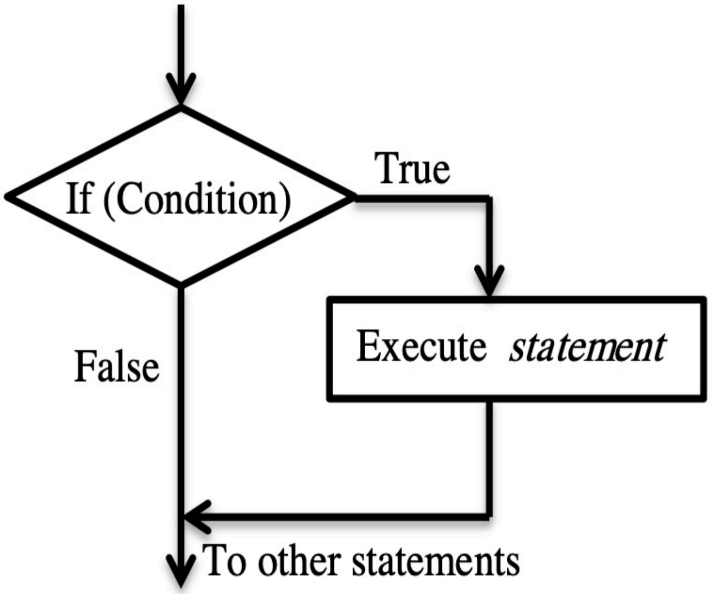
- **IF(IF-THEN) Construct**
- **IF-ELSE Construct**
- **IF-ELSEIF Construct**
- **SELECT CASE Construct**

2.1 The Simple IF Construct

The simple **IF** statement has the following form:

IF *(a simple or compound condition) statement*

If the condition is **.TRUR.** statement is executed



2.2 The Block IF Construct

If more than one statement should be executed inside the **IF**, then the following syntax should be used:

IF *(a simple or compound condition)* **THEN**

statement 1

statement 2

.

END IF

```
IF (MOD(x ,2)== 0)
   PRINT *, "x is even"
```

```
IF (MOD(x ,2)== 0) THEN
   PRINT *,x
   PRINT *,"x is even"
END IF
```

2.3 The IF-ELSE Construct

This statement is used when we have two choices . The **IF-ELSE** statement has the following form:

IF *(a simple or compound condition)* **THEN**

statement sequence_1

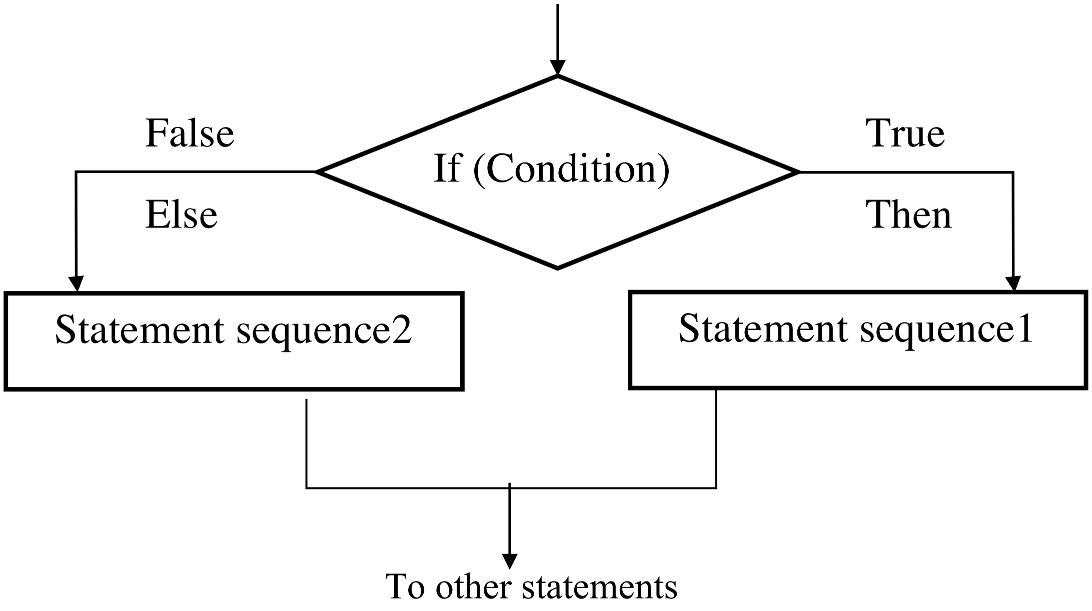
.

ELSE

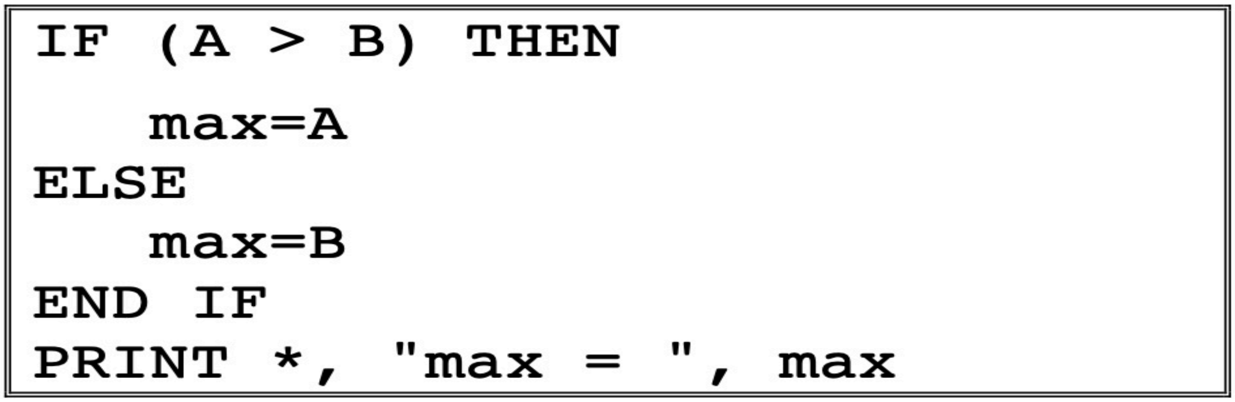
statement sequence_2

.

END IF

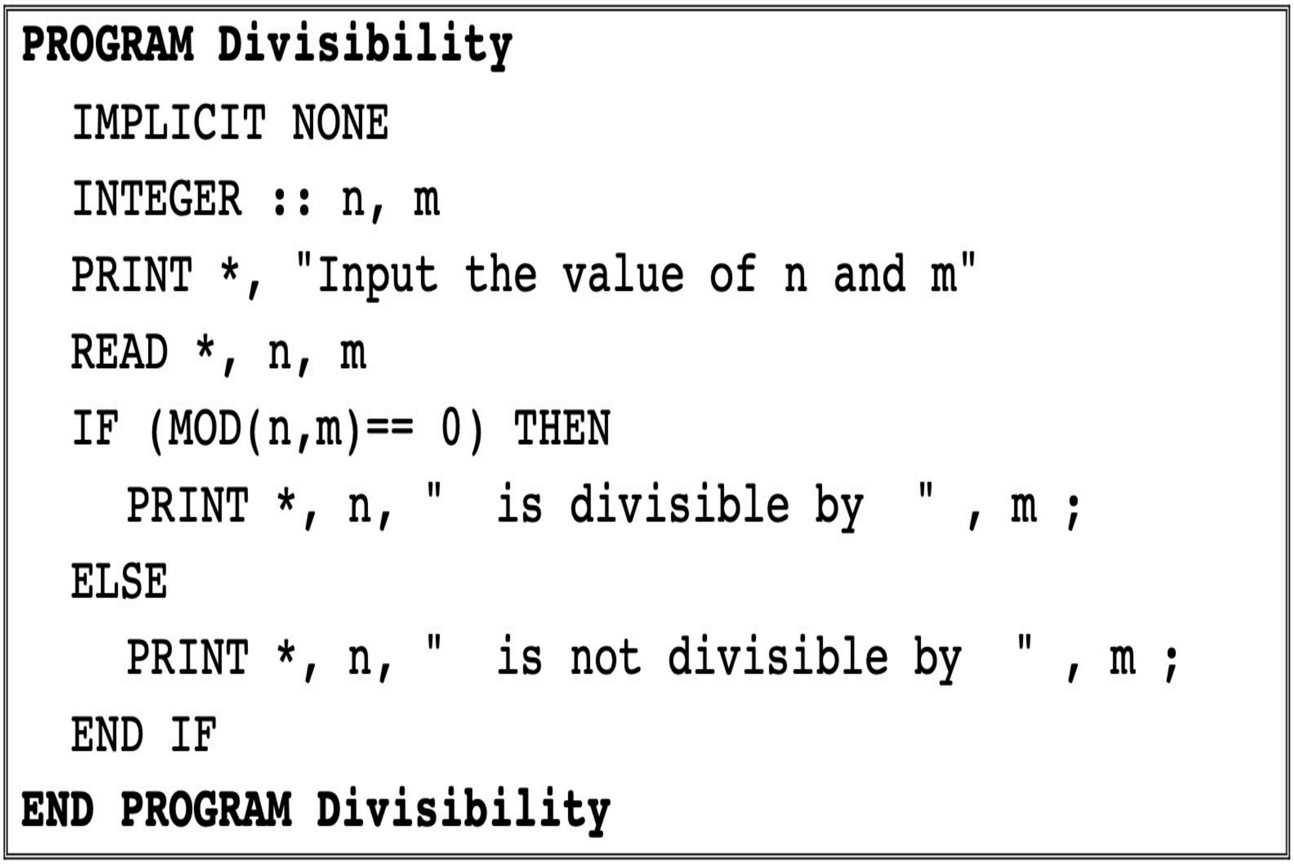


For example:



```
IF (A > B) THEN
   max=A
ELSE
   max=B
END IF
PRINT *, "max = ", max
```

The following program uses the IF-ELSE to test for divisibility (قابلية القسمة) of an integer.



```
PROGRAM Divisibility
  IMPLICIT NONE
  INTEGER :: n, m
  PRINT *, "Input the value of n and m"
  READ *, n, m
  IF (MOD(n,m)== 0) THEN
    PRINT *, n, "  is divisible by  " , m ;
  ELSE
    PRINT *, n, "  is not divisible by  " , m ;
  END IF
END PROGRAM Divisibility
```

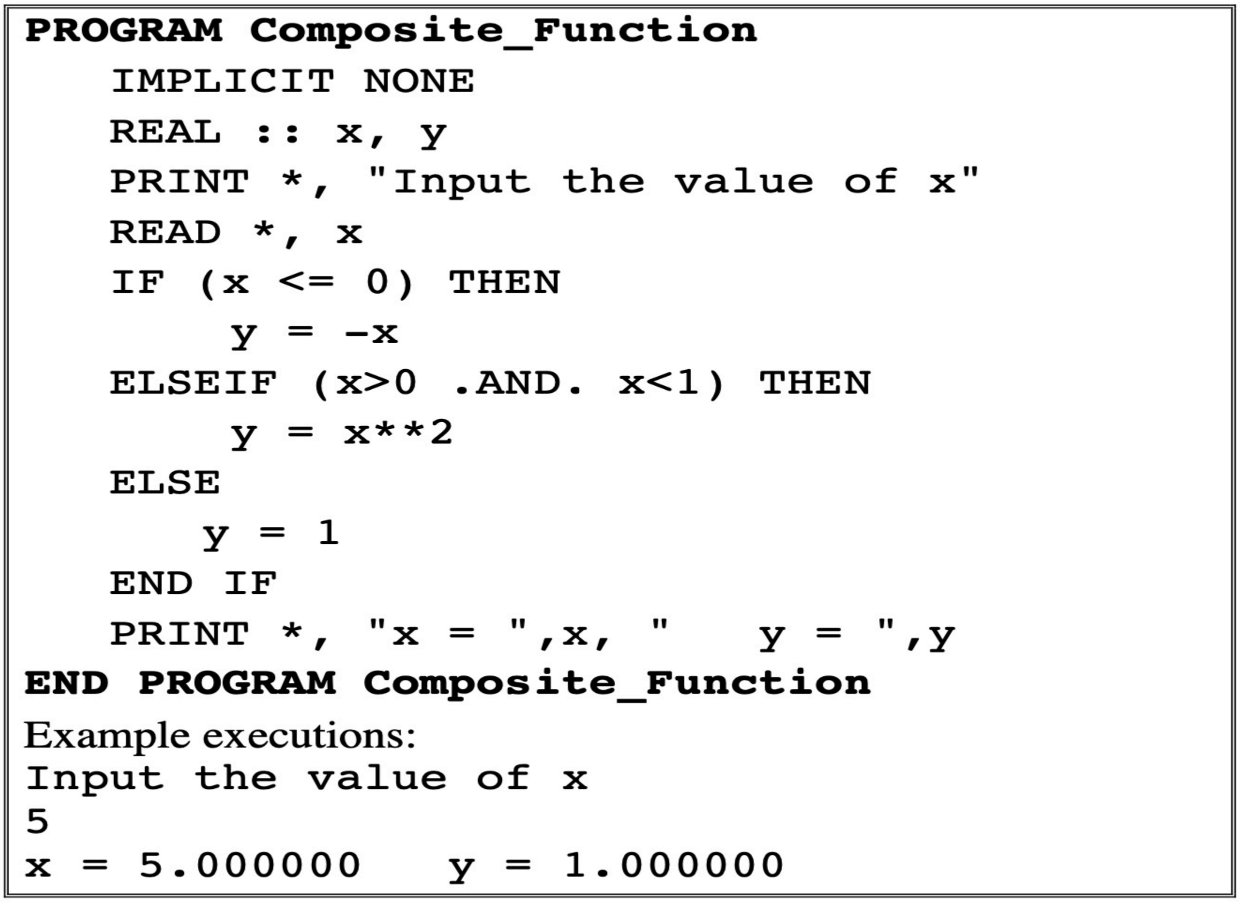
2.4 The IF-ELSEIF Construct

It is also possible to use the **IF** construct to design selection structures that contain more than two alternatives (choices):

```
IF ( a simple or compound condition) THEN
    statement sequence_1
ELSEIF ( a simple or compound condition) THEN
    statement sequence_2
    .
    .
ELSEIF ( a simple or compound condition) THEN
     statement sequence_n-1
ELSE
    statement sequence_n
END IF
```

Ex: Write a program to evaluate the following function:

$$y = \begin{cases} -x & x \le 0 \\ x^2 & 0 < x < 1 \\ 1 & x \ge 1 \end{cases}$$

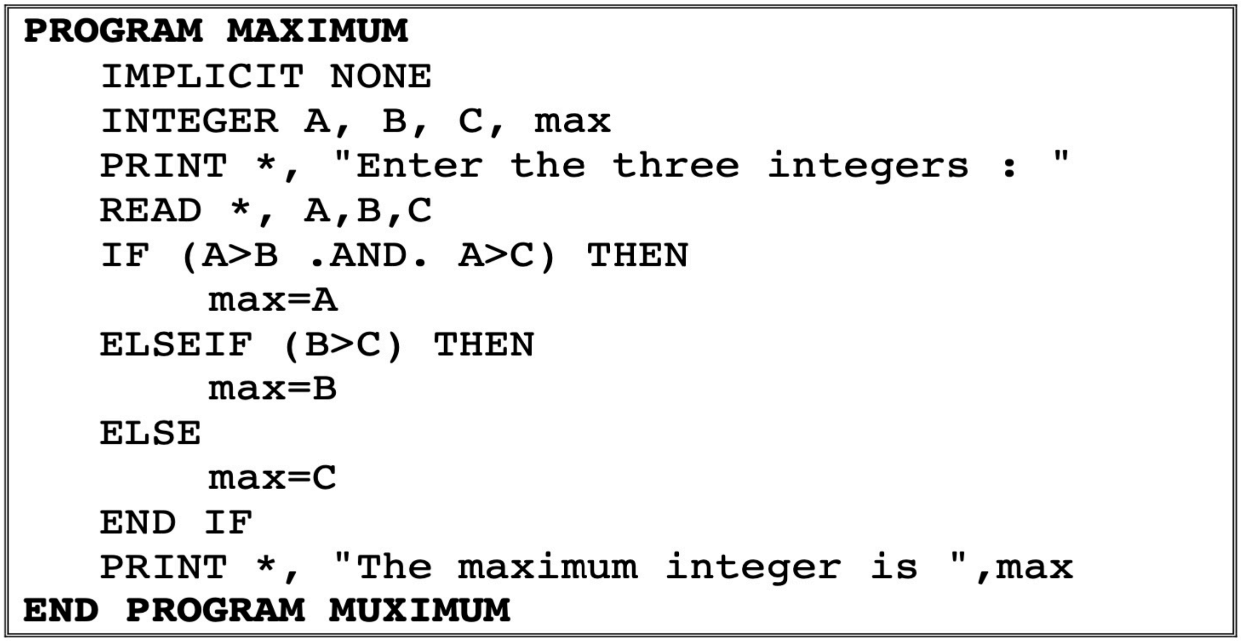


```
PROGRAM Composite_Function
   IMPLICIT NONE
   REAL :: x, y
   PRINT *, "Input the value of x"
   READ *, x
   IF (x <= 0) THEN
       y = -x
   ELSEIF (x>0 .AND. x<1) THEN
       y = x**2
   ELSE
      y = 1
   END IF
   PRINT *, "x = ",x, "   y = ",y
END PROGRAM Composite_Function
```

Example executions:

```
Input the value of x
5
x = 5.000000   y = 1.000000
```

Ex: Write a program to find the maximum of three integers.



```
PROGRAM MAXIMUM
   IMPLICIT NONE
   INTEGER A, B, C, max
   PRINT *, "Enter the three integers : "
   READ *, A,B,C
   IF (A>B .AND. A>C) THEN
       max=A
   ELSEIF (B>C) THEN
       max=B
   ELSE
       max=C
   END IF
   PRINT *, "The maximum integer is ",max
END PROGRAM MUXIMUM
```

2.5 SELECT CASE Construct

The **SELECT CASE** construct is an alternative of **IF-ELSEIF** construct and useful for implementing some selection structures. A **SELECT CASE** construct has the following form:

```
SELECT CASE ( selector )
    CASE (value 1)
        block of statement_1
    CASE (value 2)
        block of statement_2
        .
        .
    CASE (value n)
         Block of statement_n
    CASE DEFAULT
        Block of statements
END SELECT
```

Where :

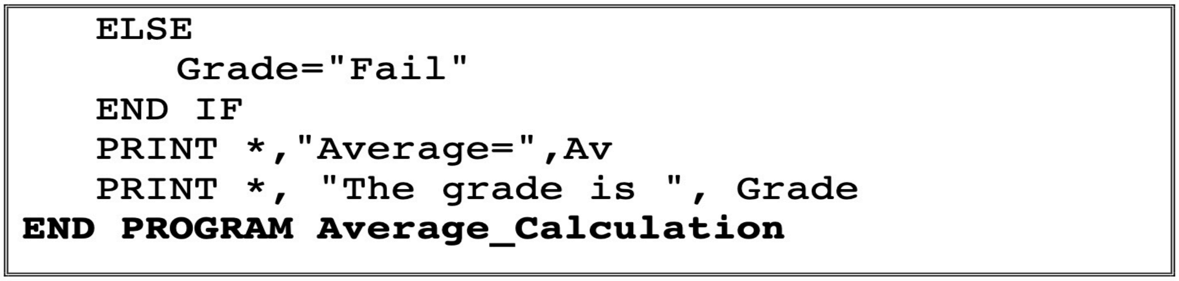
selector : is an integer, character or logical expression. It cannot be real value. value1 , value 2, ..., value n : are the possible values of the selector. The **CASE DEFAULT** block is executed if the value does not match any of the selectors.

Ex: Write a program to receive an arithmetic operator and two real numbers, the program performs the arithmetic operation on the two numbers, (use **SELECT CASE** statement).

```
PROGRAM ARITHMETIC
   IMPLICIT NONE
   CHARACTER :: ch
   REAL :: x, y
   PRINT *, "Enter the arithmetic operator : "
   READ *, ch
   PRINT *, "Enter the two numbers : "
   READ *, x, y
   SELECT CASE (ch)
     CASE ('+')
        PRINT *, x+y
     CASE ('-')
        PRINT *, x-y
     CASE('*')
        PRINT *, x*y
     CASE('/')
        PRINT *, x/y
     CASE DEFAULT
        PRINT *, "No arithmetic operation"
   END SELECT
END PROGRAM ARITHMETIC
```

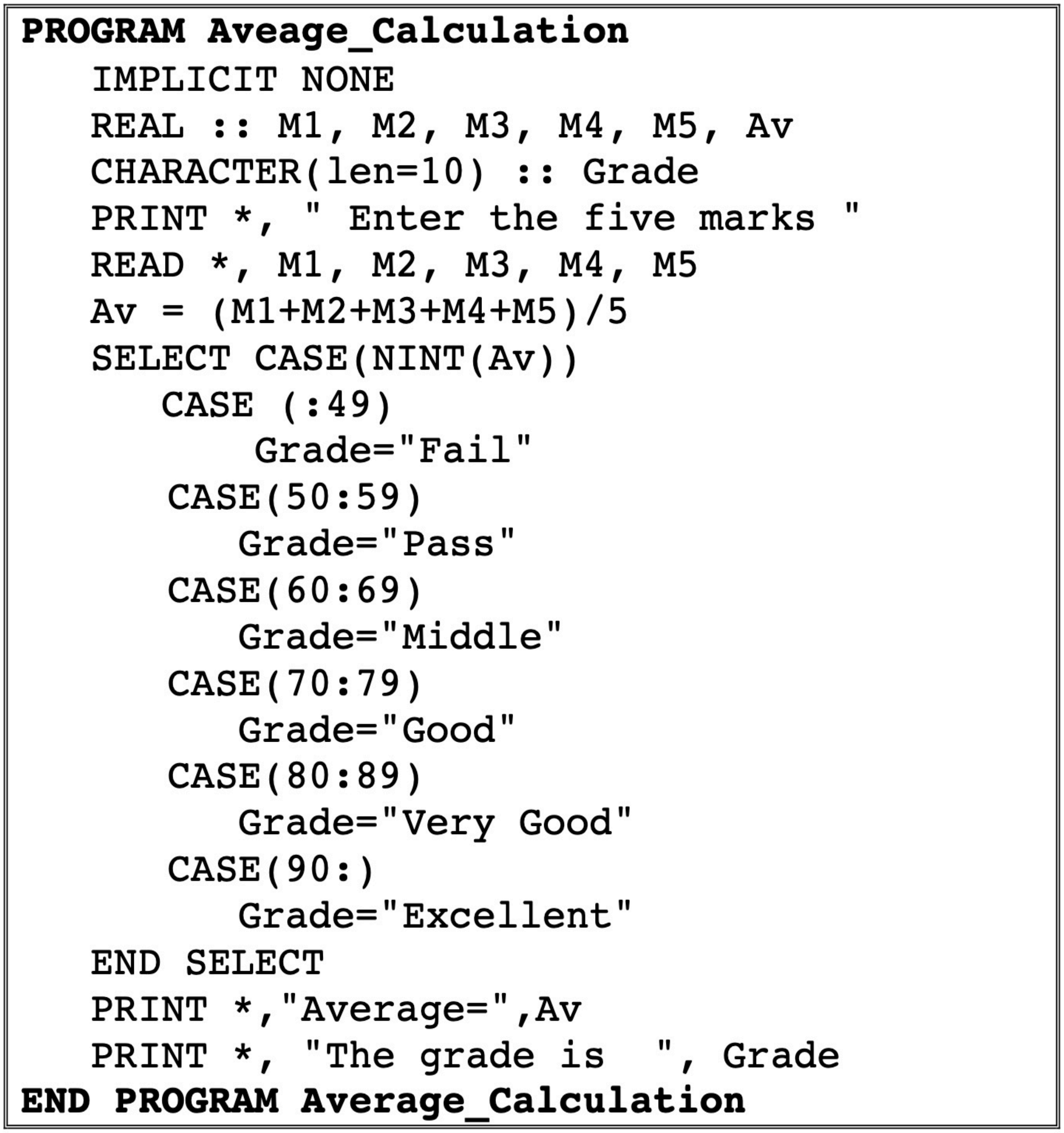
Ex: Write a program to find the average of five marks and prints the grade.

```
PROGRAM Average_Calculation
   IMPLICIT NONE
   REAL :: M1, M2, M3, M4, M5, Av
   CHARACTER(len=10) :: Grade
   PRINT *, "Enter the five marks "
   READ *, M1, M2, M3, M4, M5
   Av = (M1+M2+M3+M4+M5)/5
   IF (Av >= 50 .AND. Av < 60) THEN
      Grade="Pass"
   ELSEIF (Av >= 60 .AND. Av < 70) THEN
      Grade="Median"
   ELSEIF (Av >= 70 .AND. Av < 80) THEN
      Grade="Good"
   ELSEIF (Av >= 80 .AND. Av < 90) THEN
      Grade="Very Good"
   ELSEIF (Av >= 90) THEN
      Grade="Excellent"
```



```
   ELSE
      Grade="Fail"
   END IF
   PRINT *,"Average=",Av
   PRINT *, "The grade is ", Grade
END PROGRAM Average_Calculation
```

Ex: Re-write the above program using **SELECT CASE** statement.



```
PROGRAM Aveage_Calculation
   IMPLICIT NONE
   REAL :: M1, M2, M3, M4, M5, Av
   CHARACTER(len=10) :: Grade
   PRINT *, " Enter the five marks "
   READ *, M1, M2, M3, M4, M5
   Av = (M1+M2+M3+M4+M5)/5
   SELECT CASE(NINT(Av))
      CASE (:49)
          Grade="Fail"
      CASE(50:59)
         Grade="Pass"
      CASE(60:69)
         Grade="Middle"
      CASE(70:79)
         Grade="Good"
      CASE(80:89)
         Grade="Very Good"
      CASE(90:)
         Grade="Excellent"
   END SELECT
   PRINT *,"Average=",Av
   PRINT *, "The grade is  ", Grade
END PROGRAM Average_Calculation
```

Ex: The following program is to test a given character:

```
PROGRAM  Character_Testing
  IMPLICIT  NONE
  CHARACTER :: ch
  READ *, ch
  SELECT CASE (ch)
  CASE ('A' : 'Z' , 'a' : 'z')
       PRINT *, ch ," is a Letter"
  CASE ('0' : '9')
    `   PRINT *, ch ," is a digit"
  CASE DEFAULT
       PRINT *, ch ," is a symbol"
   END SELECT
END PROGRAM  Character_Testing
```

H.W. Write a program to find the value of y from the following .

1. using **SELECT CASE** 2. using **IF-ELSEIF**

$$y = \begin{cases} \sqrt{(x+5)^3} & x = -1 \\ 2x^3 + 5x + 4 & x = 0 \\ \mathrm{x} - \sin(\mathrm{x}) & \mathrm{x} = 1 \\ 5 & otherwise \end{cases}$$

H.W. Write a program to find the maximum of four integers