switch Statement

switch statement is used to execute a block of statements depending on the value or an expression.

The general syntax of switch statement is

```
switch (expression or variable)
{
        case <value 1> :
                        {
                                <statement block 1>
                                break;
                        }
        case <value 2> :
                        {
                                <statement block 2>
                                break;
                        }
                        .
                        .
                        .
                default:
                        {
                                <default statement block >
                                break;
                        }
}
```

Where < expression> refers any **int** or **char** expression or variable.
<**value 1**>, <**value 2**>, . . . are values which will match with the value of the expression.
break is a statement which will transfer the control to the end of switch statement.

Eg.
```
switch (sex)
{
  case 'm':   {
                male++;
                total++;
```

```
                    break;
                    }
 case 'f':          {
                    female++;
                    total++;
                    break;
                    }
  default:
        printf("\ n Error in sex code");
        break;
}
```

Ex: What is the purpose of a **switch** statement?

Solution: The purpose of a **switch** statement is to execute a block of statements depending on the value of an integer or character variable. It can also be used to replace a complicated nested **if** statement.

Ex: In what ways does a **switch** statement differ from an **if** statement?

Solution: An **if** statement executes a statement block depending on the value of a condition. A nested if statement can be used to implement a complicated condition. A **switch** statement executes a statement block depending on value of an integer or char variable.

Ex. In what situation a switch statement is used? Explain with an example.

Answer: In real life we are often faced with situations where we are required to make a choice between a number of alternatives rather than only one or two. For example, which school to join or which hotel to visit or still harder which girl to marry (we almost always end up making a wrong decision is a different matter altogether !). The C programming is same; the choice we are asked to make is more complicated then merely selecting between two alternatives. C provides a special control statement called switch statement that allows us to handle such cases effectively; rather than using a series of if statements.

Instead of using the if-else-if ladder, the switch statement is available in programming language C for handling multiple choices. The general form of the switch construct is:

```
switch(variable)
 {
        case constant1:
                statement(s);
                break;
        case constant2:
                statement(s);
                break;
        case constant3:
                statement(s);
                break;
        ……………
        default:
                statement(s);
}
```

Consider the following program:

```
main()
{
        int i=2;
        switch(i);
        {
                case 1:
                        printf(“I am in case 1\n”);
```

```
                break;
        case 2:
                printf("I am in case 2\n");
                break;
        case 3:
                printf("I ammin case 3\n");
                break;
        default:
                printf("I am in default\n");
    }
}
```

The output of this program would be :

```
I am in case 2
```

Ex: What is the use of case statement? Can it be a replacement of the if statement? Justify your answer.

Solution: The case statement is used with switch statement. Instead of using if-else, the switch statement is available in programming language C for handling multiple choices.

We have seen that when one of the many alternatives is to be selected, we can design a program using if statements to control the selection. However, the complexity of such a program increases dramatically when the number of alternatives increases. The program

becomes difficult to read and follow. At times, it may confuse even the person who designed it. Fortunately, C has a built-in multiway decision statement known as a switch. The switch statement tests the value of a given variable (or expression) against a list of case values and when a match is found; a block of statement associated with that case is executed. The general form of the switch statement is:

```
switch (variable)
{
    case constant1:
            statement(s);
            break;
    case constant2:
            statement(s);
            break;
    …….
    ……
    default:
            statement(s);
            break;
}
```

Ex: Write a C program to find the value of y using:

$$y(x,n) = \begin{cases} 1+x & \text{when } n = 1 \\ 1+x/n & \text{when } n = 2 \\ 1+x^n & \text{when } n = 3 \\ 1+nx & \text{when } n > 3 \text{ or } n < 1 \end{cases}$$

Solution:

```
/*Program to find the value of y */
```

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
main( )
{
        int n;
        float x,y;
        clrscr( );
        printf("\n Enter value to x and n: " );
        scanf("%f  %d", &x, &n);
        switch(n)
        {
                case 1: y = 1+x;
                        break;
                case 2: y = 1+x/n;
                        break;
                case 3: y = 1 + pow(x, n);
                        break;
                default: y = 1 + n * x;
                        break;
        }
        printf("\n Value of y(x, n) = %6.2f",y);
        getch( );
}
```

Q.5: Write a program to find the roots of the quadratic equation $ax^2+bx+c=0$ when the roots are:

(i) real and unequal
(ii) real and equal
(iii) complex, in which case the roots should be printed out as

$x_1=p+iq$
$x_2=p-iq$

Solution:
The program is given below:

```
/* Roots of Quadratic Equation */
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
	float a,b,c,real,num,imag,root1,root2,disc;
	int k;
	clrscr();
	printf("Input a, b, c\n");
	scanf("%f%f%f",&a,&b,&c);
	printf("%6.2f%6.2f%6.2f\n",a,b,c);
	if(a!=0)
	{
		disc=b*b-4*a*c;
		printf("Discriminant=%5.2f\n",disc);
		if(disc<0) k=1;
		if(disc==0) k=2;
		if(disc>0) k=3;
		switch(k) {
		case 1:
			printf("Roots are complex/imaginary\n");
			real=-b/(2*a);
			disc=-disc;
			num=pow((double)disc,(double)0.5);
			imag=num/(2*a);
			printf("Root1=%5.2f+j%5.2f\n",real,imag);
			printf("Root2=%5.2f-j%5.2f\n",real,imag);
			break;
		case 2:
			printf("Roots are real and equal\n");
			root1=-b/(2*a);
			printf("Root1=Root2=%7.2f\n",root1);
			break;
		case 3:
			printf("Roots are real and unequal\n");
			root1=(-b+sqrt((double)disc))/(2*a);
			root2=(-b-sqrt((double)disc))/(2*a);
			printf("Root1=%7.2f  Root2=%7.2f\n",root1,root2);
			break;
		default:
			break;
		} /* End of switch */
```

```
                }
                else
                        printf("Equation is linear\n");

        printf("\n\aPress any key to exit...");
        getch();
        return;
}
```

The output will be:

Run1:

```
Input a, b, c
  1.00  2.00  1.00
Discriminant= 0.00
Roots are real and equal
Root1=Root2=  -1.00

Press any key to exit...
```

Run2:

```
Input a, b, c
  1.00  4.00  1.00
Discriminant=12.00
Roots are real and unequal
Root1=  -0.27  Root2=  -3.73

Press any key to exit...
```

Run3:

```
Input a, b, c
  1.00  2.00  7.00
Discriminant=-24.00
Roots are complex/imaginary
Root1=-1.00+j 2.45
Root2=-1.00-j 2.45

Press any key to exit...
```

Run4:

```
Input a, b, c
  0.00  4.00  7.00
Equation is linear
```

Press any key to exit...

Ex: When is a switch statement better than multiple if statements?

Solution: A switch statement is generally best to use when you have more than two conditional expressions based on a single variable of numeric type.

&&&&&

Q.3. Write a program to input a digit/number and print it in words.
Solution:

```
/* Converting digits of a number to corresponding words */
#include<stdio.h>
main()
{
  int digit[10],num,i,j,k;
  printf("Enter number with digits from 0 to 9 (less than 4 digits): ");
  scanf("%d",&num);
  printf("%d\n",num);
  i=0;
  while(num !=0)
  {
    digit[i++]=num-(num/10)*10;
    num=num/10;
  }
  j=--i;
```

```
  printf("\nThe number in words is: ");
  for(k=j;k>=0;k--)
  switch(digit[k])
  {
    case 0:
      printf("zero ");
      break;
    case 1:
      printf("One ");
      break;
    case 2:
      printf("Two ");
      break;
    case 3:
      printf("Three ");
      break;
    case 4:
      printf("Four ");
      break;
    case 5:
      printf("Five ");
      break;
    case 6:
      printf("Six ");
      break;
    case 7:
      printf("Seven ");
      break;
    case 8:
      printf("Eight ");
      break;
    case 9:
      printf("Nine ");
      break;
  }
}
```

Run:

Enter number with digits from 0 to 9 (less than 4 digits): 3729

The number in words is: Three Seven Two Nine

Q.4: Write a menu driven program which has following options:

1. Factorial of a number.
2. Prime or not.
3. Odd or even
4. Exit

Make use of switch statement.

Solution:

```
/* Menu driven program */
#include<stdio.h>
#include<conio.h>

main()
{
        int choice,num,i;
        unsigned long int fact;
        clrscr();
        while(1)
        {
                printf("\n\n1.Factorial\n");
                printf("2.Prime\n");
                printf("3.Odd/Even\n");
                printf("4. Exit\n");
                printf("\n Your choice ?");
                scanf("%d",&choice);
                switch(choice)
                {
                        case 1:
                                printf("\nEnter number ");
                                scanf("%d",&num);
                                fact=1;
                                for(i=1;i<=num;i++)
                                        fact*=i;
                                printf("\nFactorial value=%lu",fact);
                                break;
                        case 2:
                        printf("\nEnter number ");
                        scanf("%d",&num);
                        for(i=2;i<num;i++)
                        {
                                if(num%i= =0)
                                {
                                printf("\nThe number %d is not a prime number",num);
                                        break; /*Takes control out of for loop */
                                }
                        }
                        if(i= =num)
                                printf("\nThe number %d is a Prime number",num);
                                break;
                        case 3: printf("\n Enter number ");
                                scanf("%d",&num);
                                if(num%2= =0)
                                        printf("\nThe number %d is a Even Number.",num);
                        else
                                printf("\nThe number %d is a Odd Number.",num);
                        break;
```

```
                        case 4:
                                exit(); /*Terminates program execution */
                }
        }

}
```

Q.8: Write a menu-based program to perform any two of the following operations:

(i) Check whether a number entered by the user is prime or not.
(ii) Find the factors of the given no.
(iii) Print its multiplication table.

Modularize your code using functions. **(PGDCA-July-2007/GU)**

Solution: The required program is given below:

```
/* Menu-based program*/
#include<stdio.h>
#include<conio.h>
main()
{
        int choice,num;
        void prime(int);
        void factor(int);
        void multi(int);
        clrscr();
        top:
        printf("Enter a positive number: ");
        scanf("%d",&num);
        printf("%d\n\n",num);
        if(num<0)
        {
                printf("Wrong value! :");
                goto top;
        }
        do{
        printf("********************* Menu for Choice ***************\n\n");
        printf("1. For check whether a number is prime or not.\n");
        printf("2. For finding the factors of the given no.\n");
        printf("3. For printing the multiplication table of the given no.\n");
        printf("4. Exit.\n");
        printf("*************************************************\n");
        printf("Enter your choice: ");
        scanf("%d",&choice);
        printf("%d\n",choice);
        switch(choice)
        {
                case 1:
                prime(num);
```

```
                break;
                case 2:
                factor(num);
                break;
                case 3:
                 multi(num);
            break;
            case 4:
            break; //exit(0);
                default:
                printf("Wrong choice! Enter choice again:\n\n");
                break;
        }
        }while (choice !=4);
        printf("\nPress any key to exit...\n");
        getch();
        return 0;
}
void prime(int num)
{
        int i;
        for(i=2;i<num;i++)
        {
                if(num%i= =0)
                {
                        printf("\nThe number \"%d\" is not a prime number.\n\n",num);
                        break; /*Takes control out of for loop */
                }
        }
        if(i= =num)
                printf("\nThe number \"%d\" is a Prime number.\n\n",num);
}

void factor(int num)
{
        int i=2;
        printf("\nFactors of %d are are", num);
        while(num !=1)
        {
                if(num%i= =0)
                        printf("%5d",i);
                else
                {
                        i++;
                        continue;
                }
                num=num/i;
        }
        printf("\n\n");
        return;
```

```
}
void multi(int num)
{

        int i;
        printf("The multiplication table of %d is: \n",num);
        for(i = 1; i<=10;i++)
                printf("%d X %d = %d\n",num,i,num*i);
        printf("\n");
}
```

The output of the above program will be:

```
Enter a positive number: -8

Wrong value! : Enter a positive number: 8

********** Menu for Choice **********

1. For check whether a number is prime or not.
2. For finding the factors of the given no.
3. For printing the multiplication table of the given no.
4. Exit.
**********************************************************
Enter your choice: 1

The number "8" is not a prime number.

********** Menu for Choice **********

1. For check whether a number is prime or not.
2. For finding the factors of the given no.
3. For printing the multiplication table of the given no.
4. Exit.
**********************************************************
Enter your choice: 2

Factors of 8 are are    2    2    2

********** Menu for Choice **********

1. For check whether a number is prime or not.
2. For finding the factors of the given no.
3. For printing the multiplication table of the given no.
4. Exit.
**********************************************************
Enter your choice: 3
The multiplication table of 8 is:
8 X 1 = 8
8 X 2 = 16
8 X 3 = 24
```

```
8 X 4 = 32
8 X 5 = 40
8 X 6 = 48
8 X 7 = 56
8 X 8 = 64
8 X 9 = 72
8 X 10 = 80

********** Menu for Choice **********

1. For check whether a number is prime or not.
2. For finding the factors of the given no.
3. For printing the multiplication table of the given no.
4. Exit.
**********************************************************
Enter your choice: 6
Wrong choice! Enter choice again:

********** Menu for Choice **********

1. For check whether a number is prime or not.
2. For finding the factors of the given no.
3. For printing the multiplication table of the given no.
4. Exit.
**********************************************************
Enter your choice: 4

Press any key to exit...
```

Q.9: What is the difference between 'break' and 'continue' statements used in C? Explain a suitable example.

Solution: The difference between 'break' and 'continue' statements: Whereas ***break*** terminates a loop entirely, ***continue*** merely terminates the current iteration of the loop, allowing execution to continue with the next iteration (if any). Also, ***continue*** cannot be used to terminate a ***switch*** statement.

Example: A **continue** is usually associated with an **if**. As an example, let's consider the following program.

```
main( )
{
    int i, j;
    for(i=1;i<=2;i++)
```

```
    {
        for(j=1;j<=2;j++)
        {
            if(i= =j)
                continue;
            printf("\n%d %d\n",i,j);
        }
    }
}
```

The output of the above program would be:

```
1   2
2   1
```

Note that when the value of **i** equals that of **j**, the **continue** statement takes the control to the for loop(inner) bypassing rest of the statements pending execution in the **for** loop(inner).

Example: Following program segment is written to explain the use of break with while loop structure:

```
#include<stdio.h>
main( )
{
    int i, value;
    i=0;
    while(i<=10)
    {
        printf("Enter a number\n");
        scanf("%d",&value);
        if value = =0 || value<0)
        {
            printf("Zero or negative value found.\n");
            break;
        }
        i++;
    }
}
```

The above program segment processes only the positive integers. Whenever the zero or negative value is found the program will display the message "Zero or negative value found" as an error and it will not execute the loop further.

Q.10: What is the effect of absence of break in switch-case statement? What is purpose of default?

Solution: In a switch- case statement, when a match is found, the statement sequence associated with that case is executed until a break statement or the end of switch statement is reached. So if break statements are missing the statement sequence is executed until the end of the switch-case statement is reached.

The default statement gets executed when no match is found against the values specified in the case statements of switch.