Chapter : 4
Functions

Examples

1. An example for functions.[Pg-94]

```
# include <iostream.h>
# include <conio.h>
int fact(int num)
{
 int factorial = 1;
 for(inta=1;a<=num;a++)
   factorial *= a;
 return factorial;
}
void main();
{
 int x,f;
clrscr();
cout<<"\n Enter the values…";
cin>>x>>f;
cout<<fact(x)/fact(f);
}
```

Output:

If the values are 4 and 5,

```
Enter values...4 5
2880
```

2.Function prototyping:[Pg-96]

```
# include <iostream.h>
# include <conio.h>
void fun(char name[]);
void main()
{
 char n[]={"C++ programming.."};
 fun(n);
}
void fun(char name[])
{cout<<name;}
```

Output:

```
C++ programming..
```

3.Calling function:[Pg-98]

```
# include <iostream.h>
# include <conio.h>
int add(int a,int b)
{return a+b;}
void main()
{
int x1,x2,sum=0;
cin>>x1>>x2;
sum=add(x1,x2);
cout<<sum;
}
```

Output:

If the numbers are 5 and 2,

```
5 2
7
```

4.Calling by value: [Pg-100]

```
# include <iostream.h>
# include <conio.h>
# include<iomanip.h>
void swap(int n1,int n2)
{
int temp;
temp=n1;
n1=n2;
n2=temp;
cout<<'\n'<<n1<<'\t'<<n2<<'\n';
}
void main();
{
int m1=10,m2=20;
clrscr();
cout<<"\n Values before invoking
   swap"<<m1<<'\t'<<m2;
cout<<"\n Calling swap..";
swap(m1,m2);
cout<<"\n Back to main..Values are"
     <<m1<<'\t'<<m2;
getch();
}
```

Output:

```
Values before invoking swap  10   20
Calling swap..
20   10
Back to main..Values are 10   20
```

5.Calling by reference:[Pg-101]

```
# include <iostream.h>
# include <conio.h>
# include<iomanip.h>
void swap(int &n1,int &n2)
{
int temp;
temp=n1;
n1=n2;
n2=temp;
cout<<'\n'<<n1<<'\t'<<n2<<'\n';
}
void main();
{
int m1=10,m2=20;
clrscr();
cout<<"\n Values before invoking
```

COPYRIGHT. PAUL'S

```
swap"<<m1<<'\t'<<m2;
cout<<"\n Calling swap..";
swap(m1,m2);
cout<<"\n Back to main..Values are"
     <<m1<<'\t'<<m2;
getch();
}
```

Output:

```
Values before invoking swap  10   20
Calling swap..
20   10
Back to main..Values are 20   10
```

Significance:

The difference between the program no.4 and program no.5 is that the modifications made to the formal parameters is reflected back to the actual parameters in program no.5 but it is not so in program no.4

6. Call by reference: [Pg-103]

```
# include <iostream.h>
# include <conio.h>
void main()
{
int num1=10,&num2=num1;
num2++;
cout<<num1;
getch();
}
```

Output:

```
11
```

7. Call by reference:[Pg-105]

```
# include <iostream.h>
# include <conio.h>
void fun starts(int &i)
{
int i,j=5;
for(i=1;i<=j;i++)
  cout<<"<<'*';
}
void main()
{
int mi=1;
clrscr();
for(;mi<=5;mi++)
{
 cout<<'\n';
 fun_starts(mi);
}
getch();}
```

```
*****
```

Output:

Reason:

Variable i is referenced to mi and so when i gets a value 6 then mi is also updated to 6 and the for loop of the main() is executed only once.

8.Assigning default value : [Pg-106]

```
# include <iostream.h>
# include <conio.h>
float power(float n,int p=1)
{
float prd = 1;
for(int i=1;i<=p;i++)
  prd *= n;
return prd;
}
void main()
{
clrscr();
int x=4,b=2;
cout<<"\n Call statement is
power(b,x)..."<<power(b,x);
cout<<"=n Call statement is
   power(b)..."<<power(b);
getch();
}
```

Output:

```
Call statement is power(b,x)...16
Call statement is power(b)...2
```

9. Assigning default value : [Pg-107]

```
# include <iostream.h>
# include <conio.h>
int area(int side1=10,int side2=20)
{return(side1*side2);}
void main()
{
int s1=4,s2=6;
clrscr();
cout<<area(s1,s2)<<'\n';
cout<<area(s1)<<'\n';
cout<<area(s2)<<'\n'
getch();
{
```

Output:

```
24
80
120
```

10. [Pg-108]

```
# include <iostream.h>
# include <conio.h>
void print(int times,char ch='*')
{
```

OSSST PAUL'S

```
cout<<'\n';
for(int i=1;i<=times;i++)
  cout<<ch;
}
void main()
{
clrscr();
print(50);
print('A',97);
print();
}
```

Output;

- For print(50),

'*' will be printed 50 times.

- For print('A',97),

'A' will be printed 65 times.

- For print(),

'*' will be printed 50 times.

11. Return by reference:[Pg-110]

```
# include <iostream.h>
# include <conio.h>
void main()
{
int i=5;
int &count=i;
cout<<"\n Count:"<<count;
count++;
cout<<"\n i:"<<i;
getch()
}
```

Output:

```
Count:5
i:6
```

12. [Pg-111]

```
# include <iostream.h>
# include <conio.h>
int &maxref(int &a , int &b)
{
if(a>b)
  return a;
else
  return b;
}
void main()
{
int x=20,y=30,max=0;
 max=maxref(x,y)
cout<<"\n Maximum is:"<<max;
}
```

Output;

```
Maximum is:30
```

13.[Pg-111]

```
# include <iostream.h>
# include <conio.h>
int &maxref(int &a , int &b)
{
if(a>b)
  return a;
else
  return b;
}
void main()
{
int x=20,y=30,max=0;
maxref(x,y)=-1;
cout<<"\n Value of x is:"<<x;
cout<<"\n Value of y is:"<<y;
getch();
}
```

Output:

```
Value of x is:20
Value of y is:-1
```

14. Inline functions:[Pg-113]

```
# include <iostream.h>
# include <conio.h>
inline float convert_feet(int x)
{
return x*12;

void main()
{
clrscr();
int inches=45;
cout<<convert_feet(inches);
getch();
}
```

Output:

```
540
```

15. Local scope:[Pg-114]

```
# include <iostream.h>
# include <conio.h>
void main()
{
int a=10,b=20;
if(a>b)
{
int temp;
temp=a;
a=b;
```

OSSST PAUL'S

```
b=temp;
}
cout<<"\n Descending order…";
cout<<'\n'<<a<<'\n'<<b;
getch();
}
```

Output:

```
Descending order...
10
20
```

16. Scope of local variable:[Pg-115]

```
# include <iostream.h>
# include <conio.h>
void main()
{
int a=10,b=20;
if(a>b)
{
int temp;
temp=a;
a=b;
b=temp;
}
cout<<a<<b<<temp;
getch();
}
```

Error:

```
The variable temp is not accessible.
```

The lifetime of local variable is the life time of a block in its state of execution.

17. Scope operator:[Pg-117]

```
# include <iostream.h>
# include <conio.h>
int num=15;
void main()
{
clrscr();
int num=5;
num = num+ ::num;
cout<<num<<'\t'<<++::num;
getch();
}
```

Output

```
20    16
```

Exercises: **[Pg-118]**

1.Construct function prototypes for descriptions given below:

a) procedural function()—is a function that takes no arguments and has no return values:

Solution:

```
void procedural-function(void); [or]
void procedural-function();
```

b) manipulative-function() takes 1 argument of double type and returns int type:

Solution:

```
int manipulative-function(double); [or]
int manipulative-function(double d); [or]
manipulative-function(double);
```

c)fun-default() takes two arguments,once with a default integer value,and the other float, has no return value:

Solution:

```
void fun-default(float,int num=10);
```

d)return-reference-fun() takes two int arguments and return reference to int type:

Solution:

```
int&return-reference-fun(int&,int&);
```

e)multi-arguments() tha takes two arguments of float, where the 1st argument is pi should not be modified, and the 2nd argument is fo reference type. The function has no return type.

Solution:

```
void multi-arguements(float const pi, int &a);
```

2. Identify the errors :[Pg-119]

a)float average(a,b);

Error: i.Type of variable is not declared.
ii.Semi-colon should be avoided.

Correction:

```
float average(int a, int b)
```

b)float prd(int a,b);

Error: i.Type is not declared for b.
ii.Semi-colon should be avoided.

Correction:

```
float prd(int a, int b)
```

c)int default-arg(int a=2,int b);

Error: i.Semi-colon should be avoided.

Correction:

```
int default-arg(int a=2,int b)
```

d)int fun(int, int, double=3.14);

Error: i. Variable is not declared for type double.

ii. Semi-colon should be avoided.

Correction:

```
int fun(int,int,double d=3.14)
```

e)void strings(char[]);

Error: i. Semi-colon should be avoided.

Correction:

```
void strings(char[])
```

3.Write a main() function that includes everything necessary to call this function.

void line(int times,char ch) **[Pg-119]**

```
{
char<<'\n';
for(int i=1;i<=times;i++)
cout<<ch;'
cout<<'\n';
}
```

Function program:

```
void main()
{
int times=2;
char ch='a';
line(times,ch);
}
```

4.Write the scope of all variables mentioned in this program: [Pg-120]

```
# include <iostream.h>
float a , b ; void f1(char);
int main()
{char ch;
……..
{int i=0;
……..}
}
void f1(char g)
{short x,y;}
```

Solution:

a,b → file scope

ch →function scope-main()

i →scope within its block

x,y,g →function scope-f1 function

4. Identify the errors: [Pg-120]

a) # include <iostream.h> ..1

```
xyz(int m,int n)        ..2
{int m=10;              ..3
n=m*n;                  ..4
return n;               ..5
}                       ..6
void main()             ..7
{cout<<xyz(9,27);}      ..8
```

Error:

Line:3 The variable 'm' is declared with function block,which is not permitted.

b) # include <iostream.h> ..1

```
void xyz();             ..2
void main()             ..3
{int x=xyz();}          ..4
void xyz()              ..5
{return '10';}          ..6
```

Error:

Line:6 Function declared as void type, cannot have a return statement, hence the function call cannot be a part of an expression.

Line:2 Semi-colon should be avoided for function statement.

c) # include <iostream.h> ..1

```
void counter(int &a)    ..2
{++a;}                  ..3
void main()             ..4
{counter(50);}          ..5
```

Error:

Line:5 The actual parameter cannot be passed in the form of a value,as the formal parameter is of reference type.

5.What wiil be the output of the following programs:[Pg-121]

a) # include<iostream.h>

```
int val=10;
divide(int);
void main()
(int val=5;
val=divide(::val/val);
cout<<::val<<val;
}
divide(int v)
{return v/2;}
```

OSSST PAULS

Output:

```
101
```

b)
```
# include<iostream.h>
divide(int v)
{return v/10;}
void main()
{int val=-1;
val=divide(400)==40;
cout<<"\n Val: "<<val;
}
```

Output:

```
Val: 1
```

c)
```
# include<iostream.h>
int incre(int a)
{return a++;}
void main()
{int x=10;x=incre(x);cout<<x;}
```

Output:

```
10
```

d)
```
# include<iostream.h>
void line()
{static int v=5;
int x=v--;
while(x)
{cout<<'*';x--;
}
cout<<'ln';
}
void main()
{clrscr();
for(int i=1;i<=5;i++)
  line();
getch();}
```

Output:

```
*****
****
***
**
```

e)
```
# include<iostream.h>
first(int i)
{return i++;}
second(int x)
{return x--;}
void main()
{int val=50;
val=val*val/val;
val=second(val);
val=first(val);
cout<<"\n Val: "<<val;
}
```

Output:

```
Val: 50
```

6. Program writing:[Pg-123]

a)Write a program in C++ to define a function called float cube(int,int,int).Write main() function, to test the working of cube().

```
# include <iostream.h>
# include <conio.h>
float cube(int a,int b, int c)
{
float p;
p=a*b*c;
return p;
}
void main()
{
clrscr();
float cub;
int x,y,z;
cout<<"\n Enter the numbers: ";
cin>>x>>y>>z;
cub=cube(x,y,z);
cout<<"\n Product is: "<<cub;
getch();
}
```

If numbers are 4,6,2 then

b)Define a function unsigned long int factorial(int);

The factorial of a number is calculated as follows: For example, Factorial of 5 is calculated as 1x2x3x4x5

Write a main() function to calculate the factorial(n).

```
# include <iostream.h>
# include <conio.h>
unsigned long int factorial(int n)
{
int f=1;
for(int x=1;x<=n;x++)
  f *= x;
return f;
}
void main()
{
clrscr();
int n;
cout<<"\n Enter the value: ";
cin>>n;
```

```
Enter the value: 5
Factorial value: 120
```

O S ST. PAUL'S

```
cout<<"\n Factorial value: "
   <<factorial(n)
getch();
}
```

Output:

If the value is 5 then

```
Enter the numbers: 4 6 2
Product is: 48
```

c)Define a function called as char odd_even_check(int);
The function should return 'E' if the given number is even, otherwise 'O'.Write a main() to test and execute the function odd_even_check(int) and also print relevant message.

```
# include <iostream.h>
# include <conio.h>
char odd_even_check(int n)
{
char c;
if(n%2==0)
 c='E';
else
 c='O';
return c;
}
void main()
{
clrscr();
int n;
cout<<"\n Enter the value:";
cin>>n;
if(odd_even_check(n)=='E')
 cout<<"\n Given number is even.";
else
 cout<<"\n Given number is odd.";
getch();
}
```

Output:

If the value is 9

```
Enter the value: 9
Given number is odd.
```

If the value is 6

```
Enter the value: 6
Given number is even.
```

d)Define a function int prime(int);
The function should return a value 1, if the given number is prime, otherwise -1. Write a main() to test and execute the function and also print relevant message.

```
# include <iostream.h>
# include <conio.h>
int prime(int n)
{
int pr=1,I;
for(i=2;i<=n/2;i++)
{
 if(n%i==0)
 {
  pr=-1;
  break;
 }
}
return pr;
}
void main()
{
clrscr();
int n;
cout<<"\n Enter the value: ";
cin>>n;
if(prime(n)==1)
 cout<<"\n Given number is prime.";
else
 cout<<"\n Given number is not prime.";
getch();
}
```

Output:

If the value is 5 then

If the value is 4 then.

```
Enter the value: 4
Given number is not prime.
```

```
Enter the value: 5
Given number is prime.
```

ST. PAUL'S