1. Write a c++ program to check whether given number is Palindrome or not.

```
#include<iostream.h>
#include<conio.h>

int main()
{
int n, num, digit ,rev=0; cout<<"Enter a number:";
cin>>num;
n=num;

while(num>0)
{
digit=num%10;
rev=(rev*10)+digit;
num=num/10;
}
cout<<"The reverse of the number is:"<<rev<<endl;if(n==rev)
cout<<"The number is a palindrome";else
{
cout<<"The number is not a palindrome";
}
getch();
return 0;
}
```

2. <u>**Write a c++ Program to implement Matrices multiplication**</u>

```
#include<iostream.h>
#include<conio.h>
int main()
{
int a[3][3], b[3][3], c[3][3], i, j, k;
cout<<"Enter the matrix1 values"<<endl;
for(i=0; i<3;i++)
{
for(j=0; j<3; j++)
{
cin>>a[i][j];
}
}
cout<<"Enter the matrix2 values"<<endl;
for(i=0; i<3; i++)
{
for(j=0; j<3; j++)
{
cin>>b[i][j];
}
}
for(i=0; i<3; i++)
{
for(j=0; j<3; j++)
{
c[i][j]=0;
for(k=0; k<3; k++)
{
c[i][j]=c[i][j]+a[i][k]*b[k][j];
}
}
}
cout<<"The product of two matrices is"<<endl;
for(i=0; i<3; i++)
{
for(j=0; j<3; j++)
{
cout<<c[i][j]<<" ";
}
cout<<endl;
}
getch();
return 0;}
```

3. Write a c++ program to implement Functions

```
#include<iostream.h>
#include<conio.h>

void add(int a,int b);
void sub(int a,int b);
void mul(int a,int b);
void divide(int a,int b);
void modulo(int a,int b);

int main()
{
add(20, 10);
sub(50, 30);
mul(2, 5);
divide(50, 10);
modulo(25, 4);
}
void add(int a,int b)
{
int c=a+b;
cout<<"The sum is:"<<c<<endl;
}
void sub(int a,int b)
{
int c=a-b;
cout<<"The difference is:"<<c<<endl;
}
void mul(int a,int b)
{
int c=a*b;

cout<<"The product is:"<<c<<endl;
}
void divide(int a,int b)
{
int c=a/b;
cout<<"The division is:"<<c<<endl;
}
void modulo(int a,int b)
{
int c=a%b;
cout<<"The modulo is"<<c<<endl;
getch();
}
```

4. **Write a c++ program to implement Student information Class**

```
#include<iostream.h>
#include<conio.h>
class Student
{
int rno;
float percentage;
char *name;

public:
void store(int a,float b,char *c)
{
rno = a; percentage = b;name = c;
}
void display()
{
cout<<"Roll number is:"<<rno<<endl; cout<<"Name is:"<<name<<endl;
cout<<"Percentage is:"<<percentage<<endl;
}
};

int main()
{
Student s1, s2;
cout<<"The Student 1 details are:"<<endl;s1.store (123, 88.8, "Saiteja"); s1.display();
cout<<"The Student 2 details are:"<<endl;s2.store (124, 78.8, "Raj");
s2.display();
getch();
 return 0;
}
```

5. Write a c++ program to implement Constructors.

```
#include<iostream.h>
#include<conio.h>
class employee
{
int  eid;
float salary;
public:
employee()
{
eid = 0;
salary = 0.0;
}
employee (int x,float y)
{
eid = x;
 salary = y;
}
employee (employee &e)
{
eid = e.eid;
 salary = e.salary;
}
void display()
{
cout<<"Employee is:"<<eid<<endl;
cout<<"Saalary is:"<<salary<<endl;
}
};
int main()
{
employee e1;
e1.display();
employee e2(123, 18000);
e2.display();
employee e3(e2);
e3.display();
getch();
return 0;
}
```

6. Program on friend function

```
#include<iostream.h>
#include<conio.h>
class Sample
{
private:int a; int b; public:

Sample()
{
a = 5;
b = 15;
}

friend void function1(Sample S);
};

void function1(Sample S)
{
cout<<"The private data members are:"<<endl;
cout<<S.a<<endl;
cout<<S.b<<endl;
}
int main()
{
Sample S;
function1(S);
getch();
 return 0;
}
```

7. Program on Function template

```
#include<iostream.h>
#include<conio.h>
 template<class T>
T max(T a, T b)
{
if (a>b)
return a;
else
return b;
}
template<class F>
F min(F a, F b)
{
if (a<b)
return a;
else
 return b;
}
int main()
{
int a,b;
cout<<"Enter a and b values:"<<endl;
cin>>a>>b;
cout<<"The maximum value is:"<<max(a,b)<<endl;
cout<<"The minimum value is:"<<min(a,b)<<endl;
getch();
return 0;
}
```

8. Program on multiple inheritances

```
 #include<iostream.h>
#include<conio.h>
class Student
{
protected:
int rno, m1, m2;
public:
void getdata()
{
cout<<"Enter the roll no:"; cin>>rno;
cout<<"Enter the two subjects marks:";
cin>>m1>>m2;
}
};

class sports
{
protected:
int sm;
public:
void getsm()
{
cout<<"\nEnter the sports mark:"; cin>>sm;
}
};

class statement: public Student, public sports
{
int tot,avg; public:
void display()
{
tot = (m1+m2+sm); avg = tot/3;
cout<<"\nRoll No:"<<rno; cout<<"\nTotal:"<<avg;
}
};

int main()
{
statement S;
S.getdata();
S.getsm();
S.display();
getch();
return 0;
}
```

9. Program on Single inheritance

```
#include <iostream>
using namespace std;
 class Account {
   public:
   float salary = 60000;
 };

   class Programmer: public Account
 {
   public:
   float bonus = 5000;
   };

int main(void) {
     Programmer p1;
     cout<<"Salary: "<<p1.salary<<endl;
     cout<<"Bonus: "<<p1.bonus<<endl;
    return 0;
}
```

10. Program on Exception handling

```
#include<iostream.h>
int main()
{
  try
  {
    int age=16;
    int voting_age=18;
    if (age>=voting_age)
    {
      cout<< "You are eligible for voting.";
    }
    else
    {
      throw "You are not eligible for voting.";
    }
  }

  catch (const char* msg)
  {
    cout<< "For voting, You must be at least 18 years old." <<endl;
    cout<<msg;
  }
  return 0;
}
```

11. C++ PROGRAM TO IMPLEMENT CLASS TEMPLATE

```
#include <iostream.h>

// Class template
template <typename T>
class Container {
private:
   T value;

public:
   // Constructor
   Container(T val) : value(val) {}
   // Method to set the value
   void setValue(T val) {
      value = val;
   }

   // Method to get the value
   T getValue() const {
      return value;
   }
};
int main() {
   // Create an object of type int
   Container<int> intContainer(42);
   cout << "Integer value: " << intContainer.getValue() << endl;
   // Create an object of type double
   Container<double> doubleContainer(3.14);
   cout << "Double value: " << doubleContainer.getValue() << endl;
   // Create an object of type string
   Container<string> stringContainer("Hello, Templates!");
   cout << "String value: " << stringContainer.getValue() << endl;
   return 0;
}
```

```
Integer value: 42
Double value: 3.14
String value: Hello, Templates!
```

12. C++ PROGRAM TO IMPLEMENT EXCEPTION HANDLING

```
#include <iostream.h>
int main() {
    int numerator, denominator;

    cout << "Enter numerator: ";
    cin >> numerator;

    cout << "Enter denominator: ";
    cin >> denominator;

    try {
        if (denominator == 0) {
            throw "Division by zero error!";
        }
        cout << "Result: " << numerator / denominator << endl;
    } catch (const char* e) {
        cout << "Exception: " << e << endl;
    }

    cout << "Program continues after exception handling." << endl;
    return 0;
}
```

Input:

```
Enter numerator: 10
Enter denominator: 2
```

Output:

```
Result: 5
Program continues after exception handling.
```

Input:

```
Enter numerator: 10
Enter denominator: 0
```

Output:

```
Exception: Division by zero error!
Program continues after exception handling.
```