Lecture # 14 outline and homework (21-07-2023)

Last Lecture Content:

- **Inheritance**

Today's Lecture

- **Diamond problem**
- **Virtual Inheritance**

Q.1. What is difference between code1.cpp and code2.cpp given below? Run both codes and feel the difference.

/////code1.cpp

```cpp
class Alpha
{
	int a;

public:
	Alpha(int x = 1)
	{
		cout << "\nIN Alpha class constructor\n";
		a = x;
	}
	void print()
	{
		cout << "\nIN Alpha class Print";
		cout << "\n a is :  " << a;

	}
	~Alpha()
	{
		cout << "\nIN Alpha class Destructor\n";

	}
};
class Beta
{

	int b;

public:
	Beta(int y = 1)
	{
		cout << "\nIN Beta class constructor\n";
		b = y;
	}
	void print()
	{
		cout << "\IN Beta class Print";
		cout << "\n B is :  " << b;


	}
	~Beta()
	{
		cout << "\nIN Beta class Destructor\n";

	}
};

class child : public Alpha, public Beta
{
	int c;

public:
	child(int c1 = 1, int c2 = 1, int c3 = 1):Beta(c3), Alpha(c2)
	{
		cout << "\nIN child class constructor\n";
		c = c1;
	}
	~child()
	{
		cout << "\nIN child class Destructor\n";

	}
};

int main()
{
	child c1(2, 2,2);




}
```

/////code2.cpp

```
class Alpha
{
	int a;

public:
	Alpha(int x = 1)
	{
		cout << "\nIN Alpha class constructor\n";
		a = x;
	}
	void print()
	{
		cout << "\nIN Alpha class Print";
		cout << "\n a is :  " << a;

	}
	~Alpha()
	{
		cout << "\nIN Alpha class Destructor\n";

	}
};
class Beta
{

	int b;

public:
	Beta(int y = 1)
	{
		cout << "\nIN Beta class constructor\n";
		b = y;
	}
	void print()
	{
		cout << "\IN Beta class Print";
		cout << "\n B is :  " << b;


	}
	~Beta()
	{
		cout << "\nIN Beta class Destructor\n";

	}
};

class child : public Beta, public Alpha
{
	int c;

public:
	child(int c1 = 1, int c2 = 1, int c3 = 1):Beta(c3), Alpha(c2)
	{
		cout << "\nIN child class constructor\n";
		c = c1;
	}
	~child()
	{
		cout << "\nIN child class Destructor\n";

	}
};

int main()
{
	child c1(2, 2,2);




}
```

Q.2: What is problem with the following code3.cpp? Attempt all possible solutions.

////code3.cpp

```
class Alpha
{
	int a;

public:
	Alpha(int x = 1)
	{
		cout << "\nIN Alpha class constructor\n";
		a = x;
	}
	void print()
	{
		cout << "\nIN Alpha class Print";
		cout << "\n a is :  " << a;

	}
	~Alpha()
	{
		cout << "\nIN Alpha class Destructor\n";

	}
};
class Beta
{

	int b;

public:
	Beta(int y = 1)
	{
		cout << "\nIN Beta class constructor\n";
		b = y;
	}
	void print()
	{
		cout << "\IN Beta class Print";
		cout << "\n B is :  " << b;


	}
	~Beta()
	{
		cout << "\nIN B class Destructor\n";

	}
};

class child : public Alpha, public Beta
{
	int c;

public:
	child(int c1 = 1, int c2 = 1, int c3 = 1):Beta(c3), Alpha(c2)
	{
		cout << "\nIN child class constructor\n";
		c = c1;
	}
	~child()
	{
		cout << "\nIN child class Destructor\n";

	}
};

int main()
{
	child c1(2, 2,2);

	c1.print();


}
```

Q.3: In code4.cpp have diamond problem what will be the possible solution(s).

```
///code4.cpp

class Grand
{
	int g;

public:
	Grand(int x = 10)
	{
		cout << "\nIN Grand class constructor\n";
		g = x;
	}
	void print()
	{
		cout << "\nIN Grand class Print";
		cout << "\n g is :  " << g;

	}
	~Grand()
	{
		cout << "\nIN Grand class Destructor\n";

	}
};

class Alpha: public Grand
{
	int a;

public:
	Alpha(int x = 1)
	{
		cout << "\nIN Alpha class constructor\n";
		a = x;
	}
	~Alpha()
	{
		cout << "\nIN Alpha class Destructor\n";

	}
};
class Beta : public Grand
{

	int b;

public:
	Beta(int y = 1)
	{
		cout << "\nIN Beta class constructor\n";
		b = y;
	}
	~Beta()
	{
		cout << "\nIN B class Destructor\n";

	}
};

class child : public Alpha, public Beta
{
	int c;

public:
	child(int c1 = 1, int c2 = 1, int c3 = 1):Beta(c3), Alpha(c2)
	{
		cout << "\nIN child class constructor\n";
		c = c1;
	}
	~child()
	{
		cout << "\nIN child class Destructor\n";

	}
};

int main()
{
	child c1(2, 2,2);

	c1.print();


}
```