Э.Жавхлан /20b1num0649/

Лаборатори 7:

1. Удамшил гэж юу вэ? (тодорхойлолт, C++ дээр хэрхэн хэрэгжүүлдэг талаар бичнэ)

Програмчлалд удамшил нь програмчлалын кодыг бүхлээр нь эсвэл зарим хэсгийг дахин ашиглах тухай ойлголт юм. Удамших бол классаас шинэ класс үүсгэх процесс. Удамших класс нь эх классынхаа бүх эсвэл зарим өгөгдлийг өвлөж авдаг.

```
class person{
  public:
        char name[20];
        int age ;
};
class student: public person{
  public:
        int grade;
        int credit;
};
```

2. Удамшлын горим. public, private, protected горимын талаар тайлбарлаж жишээгээр батална.

Удамшилын горим гэдэг нь эх класс хэрхэн удамшихыг заадаг ойлголт юм.

Public горимоор удамших тохиолдолд эх классын гишүүн өгөгдөл гишүүн функцийн хандалтын түвшин өөрчлөгдөхгүйгээр хүүхэд класст өвлөгдөнө. Энэ үед public болон protected гишүүд хот класст удамшиж хот классын public болон protected гишүүд болно.

```
class Country {
        Private: int a;
        Public:int b;
        Protected: float c;
}
class City:public country {
        Public:int b;
        Protected: float c;
        ...
}
```

Private горимоор удамших тохиолдолд эх классын protected болон public хандалтын түвшинтэй гишүүн өгөгдөл гишүүн функц нь удамшихдаа хүүхэд классд private хандалтын түвшинтэй болж удамшина. Тус горимоор удашихад эх классын private өгөгдөл хүүхэд классд удамшихгүй. Энэ үед public болон protected гишүүд хот класст удамшиж хот классын private гишүүд болно.

```
class Country {
        Private: int a;
        Public:int b;
        Protected: float c;
}
class City: private country {
   private:
        int b;
        float c;
   ...
}
```

Protected горимоор удамших тохиолдолд эх класст байгаа гишүүн өгөгдөл гишүүн функц Protected болж удамшина.

3. Удамшлын давуу талуудыг тоочин бичиж бодит жишээн дээр тайлбарла.

Байгаа классыг удамшуулан ашигласнаар бичих кодны хэмжээ багасхаас гадна оршин байгаа класс нь аль хэдийн туршигдсан, алдааг зассан байх учраас програмд гарч болох алдааг багасгах боломжтой.

- Классыг дахин ашиглах буюу өөр классд ашиглах.
- Нэг эх классаас олон класс үүсгэж болно.
- Байгаа классыг удамшуулан ашигласнаар програмд гарч болох алдааг багасахаас гадна программ боловсруулах хугацаа богинсох боломжтой.

4. Удамшлын хэдэн төрөл байдаг вэ? Тус бүрийг тайлбарлан бич.

Удамшлын 5 төрөл байдаг.

1. **1-1 буюу дан, энгийн удамшил:** Зөвхөн нэг эх классаас л удамшина.
2. **1-Олон буюу нийлмэл удамшил:** Удамших класс 2-оос цөөнгүй эх класстай. Жишээ нь аав ээж хоёроос хүү удамшина гэвэл аав ээж 2 нь эх классууд болно.
3. **Олон түвшинт удамшил:** Удамших класс удамшсан классаас үүсэх.

4. **Шаталсан удамшил:** Нэг эх классаас хэд хэдэн класс удамших.
5. **Холимог удамшил:** Нэг эх классаас олон хүүхэд класс удамшиж, эдгээр классуудаас шинж чанарыг нь өвлөн авч дахин шинэ класс үүсэхийг хэлнэ.

```
using namespace std;
#define pi 3.14

class Shape {
protected:
  char *name;
public:
  Shape()
  {
     name = new char[2];
     strcpy(name, "");
  }
  ~Shape()
  {
     delete name;
  }
};
class TwoDiShape : public Shape {
protected:
  int *x;
  int *y;
public:
  float area();
  float perimeter();
};
class Circle : public TwoDiShape {
private:
```

```
  float rad;
public:
  Circle()
  {
     name = new char[6];
     strcpy(name, "Circle");
     x = new int[1];
     y = new int[1];
     rad = 0;
  }
  ~Circle()
  {
     delete name;
     delete x;
     delete y;
  }
  void setX(int _x)
  {
     x = &_x;
  }
  void setY(int _y)
  {
     y = &_y;
  }
  void setRad(int r)
  {
     rad = r;
  }
  float area()
  {
     return pi * rad * rad;
  }
```

```
  float perimeter()
  {
     return 2 * pi * rad;
  }
};
class Square : public TwoDiShape {
private:
  float a;

public:
  Square()
  {
     name = new char[6];
     strcpy(name, "Square");
     x = new int[4];
     y = new int[4];
     a = 0;
  }
  ~Square()
  {
     delete name;
     delete x;
     delete y;
  }
  void setX(int _x1, int _x2, int _x3, int _x4)
  {
     x[0] = _x1;
     x[1] = _x2;
     x[2] = _x3;
     x[3] = _x4;
  }
  void setY(int _y1, int _y2, int _y3, int _y4)
```

```
  {
     y[0] = _y1;
     y[1] = _y2;
     y[2] = _y3;
     y[3] = _y4;
  }
  void setA(int side)
  {
     a = side;
  }
  float area()
  {
     return a * a;
  }
  float perimeter()
  {
     return 4 * a;
  }
};
class Triangle : public TwoDiShape {
private:
  float a;
public:
  Triangle()
  {
     name = new char[8];
     strcpy(name, "Triangle");
     x = new int[3];
     y = new int[3];
     a = 0;
  }
  ~Triangle()
```

```
  {
     delete name;
     delete x;
     delete y;
  }
  void setX(int _x1, int _x2, int _x3)
  {
     x[0] = _x1;
     x[1] = _x2;
     x[2] = _x3;
  }
  void setY(int _y1, int _y2, int _y3)
  {
     y[0] = _y1;
     y[1] = _y2;
     y[2] = _y3;
  }
  void setA(int side)
  {
     a = side;
  }
  float area()
  {
     return sqrt(3) / 4 * a * a;
  }
  float perimeter()
  {
     return 3 * a;
  }
};
```