Programlama Dillerinin Prensipleri

Lab Notları – 10

Hata Yakalama

```
public class Hesap {
   private double para;
   public Hesap(){
      para=0;
   }
   public void ParaYatir(double miktar){
      para += miktar;
   }
   public void ParaCek(double miktar){
      if(miktar > para){
         throw new IllegalArgumentException("Yeterli bakiye yok");
      }
      para = para - miktar;
   }
}
```

```
public static void main(String[] args) {
      // TODO code application logic here
      Hesap h = new Hesap();
      h.ParaCek(100);
   }
```

```
Exception in thread "main" java.lang.IllegalArgumentException: Yeterli bakiye yok
        at aaab.Hesap.ParaCek(Hesap.java:23)
        at aaab.AAAB.main(AAAB.java:21)
Java Result: 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
public void ParaCek(double miktar) throws IllegalArgumentException {
   ...
}
```

C tarafına bakıldığında, C programlama dilinde hata fırlatma ve yakalama deyimleri desteklenmez fakat buna benzer yapıyı gerçekleştirmek için jumper yapıları sunar bunlar setjmp.h kütüphanesi içerisinde bulunur. Temel mantık istenmeyen bir durumu önceden sezip programcının oraya jumper değerini değiştirecek bir kod koymasıdır. Dolayısıyla fonksiyon çalışacağı sırada değer kontrol edilirse istisnai durumun oluşması engellenip programın sonlanmasına izin verilmez.

```
#include "stdio.h"
#include "setjmp.h"
jmp_buf jumper;
int Bol(int x,int y){
        if(y == 0)
                longjmp(jumper, -3);
        return x/y;
}
int main(){
        int a=10, b=0;
        if(setjmp(jumper) == 0){
                printf("%d",Bol(a,b));
        }
```

```
        else{
                printf("Sifira Bolunme Hatasi");
        }
        return 0;
}
```

try-catch Blokları JAVA

```
...
public void ParaYatir(double miktar) throws ArithmeticException{
    if(miktar <= 0){
       throw new ArithmeticException("Çekilecek miktar sıfırdan büyük olmaldır.");
    }
    para += miktar;
  }
....
```

```
public static void main(String[] args) {
    // TODO code application logic here
    Hesap h = new Hesap();
    try{
       h.ParaCek(100);
       h.ParaYatir(-1);
    }
    catch(IllegalArgumentException ex){
       System.out.println(ex.getMessage());
    }
    catch(ArithmeticException ex){
       System.out.println(ex.getMessage());
    }
  }
```

```
// Yada tek catch bloğunda birleştirilebilir.
public static void main(String[] args) {
    // TODO code application logic here
    Hesap h = new Hesap();
    try{
       h.ParaCek(100);
       h.ParaYatir(-1);
    }
    catch(IllegalArgumentException | ArithmeticException ex){
       System.out.println(ex.getMessage());
    }
  }
```

Java'daki aynı kodun C dilinde gerçeklenmesi

```
#ifndef DOSYA_H
#define DOSYA_H

#include "stdio.h"
#include "stdlib.h"
#include "setjmp.h"

struct DOSYA{
        char *yol;
        char *icerik;
        jmp_buf jumper;
        char* (*Oku)(struct DOSYA*);
        void (*Yoket)(struct DOSYA*);
};
typedef struct DOSYA* Dosya;
```

```
Dosya DosyaOlustur(char*);
char* TumIcerikOku(const Dosya);
void DosyaYoket(Dosya);
#endif
```

```
#include "Dosya.h"

Dosya DosyaOlustur(char *yol){
        Dosya this;
        this = (Dosya)malloc(sizeof(struct DOSYA));
        this->yol=yol;
        this->icerik=NULL;
        this->Oku = &TumIcerikOku;
        this->Yoket=&DosyaYoket;
        return this;
}

char* TumIcerikOku(const Dosya d){
  char *icerik = NULL;
   int boyut = 0;
   FILE *fp;

   fp = fopen(d->yol, "r");
   if(!fp)longjmp(d->jumper,-3);
   fseek(fp, 0, SEEK_END);
   boyut = ftell(fp);
   rewind(fp);

   icerik = (char*) malloc(sizeof(char) * boyut);
   fread(icerik, 1, boyut, fp);
   fclose(fp);

   d->icerik = icerik;
   return icerik;
}

void DosyaYoket(Dosya d){
        if(d == NULL)return;
        if(d->icerik != NULL)free(d->icerik);
        free(d);
        d=NULL;
}
```

```
#include "Dosya.h"

int main(int argc, char *argv[]){
        Dosya d;
        d=DosyaOlustur("D:\\dene.txt");
        if(setjmp(d->jumper)==0){
                printf("%s",d->Oku(d));
        }
        else printf("Dosya okuma hatasi\n");
        d->Yoket(d);
        return 0;
}
```

finally Bloğu

```
public class Hesap {
   private double para;
   public Hesap(){
```

```
    para=0;
  }
  public void ParaYatir(double miktar){
    if(miktar <= 0){
      throw new ArithmeticException("Çekilecek miktar sıfırdan büyük olmaldır.");
    }
    para += miktar;
  }
  public void ParaCek(double miktar){
    if(miktar > para){
      throw new IllegalArgumentException("Yeterli bakiye yok");
    }
    para = para - miktar;
  }
  @Override
  public String toString() {
    return String.valueOf(para);
  }
}
```

```
public static void main(String[] args) {
    Hesap h = new Hesap();
    try{
      try{
        h.ParaCek(100);
      }
      catch(IllegalArgumentException ex){
        System.out.println(ex.getMessage());
        h.ParaYatir(-1);
      }
      System.out.println(h);
    }
    catch(ArithmeticException ex){
      System.out.println(ex.getMessage());
    }
  }
```

Eğer bakiyenin yazdırıldığı (koyu renkli satır) finally bloğu içerisine alınsaydı hata fırlatılsın ya da fırlatılmasın ekrana yazdırılacaktı.

```
…
finally{
   System.out.println(h);
}
…
```

Kendi Exception (Hata) Sınıfını Tasarlamak JAVA

```
public class YetersizBakiye extends IOException {
  public YetersizBakiye() { }
  public YetersizBakiye(String hataMesaji){
    super(hataMesaji);
  }
}
```

```
...
public void ParaCek(double miktar) throws YetersizBakiye{
    if(miktar > para){
      throw new YetersizBakiye("Yeterli bakiye yok");
    }
    para = para - miktar;
```

```
    }
...
```

```
public static void main(String[] args) {
      // TODO code application logic here
      Hesap h = new Hesap();
      try{
         h.ParaCek(100);
      }
      catch(YetersizBakiye ex){
         System.out.println(ex.getMessage());
      }
   }
```

Kendi Exception (Hata) Sınıf Tasarımının C dilinde Benzetimi

```
// Hata.h dosyası
#ifndef HATA_H
#define HATA_H

#include "stdio.h"
#include "stdlib.h"
#include "setjmp.h"

struct HATA{
          char *mesaj;
          jmp_buf jumper;
          char* (*getMessage)(struct HATA*);
          void (*Yoket)(struct HATA*);
};
typedef struct HATA* Hata;

Hata HataOlustur(char*);
char* GetMessage(const Hata);
void HataYoket(Hata);

#endif
```

```
//Hata.c dosyası
#include "Hata.h"

Hata HataOlustur(char *mesaj){
          Hata this;
          this = (Hata)malloc(sizeof(struct HATA));
          this->mesaj = mesaj;
          this->getMessage = &GetMessage;
          this->Yoket = &HataYoket;
          return this;
}
char* GetMessage(const Hata this){
          return this->mesaj;
}
void HataYoket(Hata this){
          if(this == NULL) return;
          free(this);
}
```

```
// DosyaOkumaHatasi.h dosyası
#ifndef DOSYAOKUMAHATASI_H
#define DOSYAOKUMAHATASI_H

#include "Hata.h"

struct DOSYAOKUMAHATASI{
	Hata super;
	char* (*getMessage)(struct DOSYAOKUMAHATASI*);
	void (*Yoket)(struct DOSYAOKUMAHATASI*);
};
typedef struct DOSYAOKUMAHATASI* DosyaOkumaHatasi;

DosyaOkumaHatasi DosyaOkumaHatasiOlustur(char*);
char* MesajGetir(const DosyaOkumaHatasi);
void throw(const DosyaOkumaHatasi);
void DosyaOkumaHatasiYoket(DosyaOkumaHatasi);
#endif
```

```
// DosyaOkumaHatasi.c dosyası
#include "DosyaOkumaHatasi.h"

DosyaOkumaHatasi DosyaOkumaHatasiOlustur(char *mesaj){
	DosyaOkumaHatasi this;
	this = (DosyaOkumaHatasi)malloc(sizeof(struct DOSYAOKUMAHATASI));
	this->super = HataOlustur(mesaj);
	this->getMessage = &MesajGetir;
	this->Yoket = &DosyaOkumaHatasiYoket;
	return this;
}
char* MesajGetir(const DosyaOkumaHatasi this){
	return this->super->getMessage(this->super);
}
// yapı ile alakalı olmayan ayrı bir metot
// Bu hata için belirlenen değer -3
void throw(const DosyaOkumaHatasi hata){
	longjmp(hata->super->jumper,-3);
}
void DosyaOkumaHatasiYoket(DosyaOkumaHatasi this){
	if(this == NULL) return;
	this->super->Yoket(this->super);
	free(this);
}
```

```
//Dosya.h  yeni hali
#ifndef DOSYA_H
#define DOSYA_H

#include "stdio.h"
#include "stdlib.h"

#include "DosyaOkumaHatasi.h"


struct DOSYA{
```

```
        char *yol;
        char *icerik;
        DosyaOkumaHatasi okumaHatasi;

        char* (*Oku)(struct DOSYA*);
        void (*Yoket)(struct DOSYA*);
};
typedef struct DOSYA* Dosya;

Dosya DosyaOlustur(char*);
char* TumIcerikOku(const Dosya);
void DosyaYoket(Dosya);

#endif
```

```
// Dosya.c yeni hali
#include "Dosya.h"

Dosya DosyaOlustur(char *yol){
        Dosya this;
        this = (Dosya)malloc(sizeof(struct DOSYA));
        this->yol = yol;
        this->icerik = NULL;
        this->okumaHatasi = DosyaOkumaHatasiOlustur("Dosya Okuma Hatasi");

        this->Oku = &TumIcerikOku;
        this->Yoket = &DosyaYoket;
        return this;
}
char* TumIcerikOku(const Dosya this){
        char* icerik = NULL;
        int boyut=0;
        FILE *fp;

        fp = fopen(this->yol,"r");
        if(!fp) throw(this->okumaHatasi);
        fseek(fp,0,SEEK_END);
        boyut = ftell(fp);
        rewind(fp);
        icerik = (char*)malloc(sizeof(char)*boyut);
        fread(icerik,1,boyut,fp);
        fclose(fp);
        this->icerik = icerik;
        return icerik;
}
void DosyaYoket(Dosya this){
        if(this == NULL) return;
        if(this->icerik != NULL) free(this->icerik);
        this->okumaHatasi->Yoket(this->okumaHatasi);
        free(this);
}
```

```
// Test.c dosyası
#include "Dosya.h"
```

```
int main(){
	Dosya dosya = DosyaOlustur("Deneme.txt");
	if(setjmp(dosya->okumaHatasi->super->jumper) == 0){
		char* str = dosya->Oku(dosya);
		printf("%s",str);
	}
	else printf("%s",dosya->okumaHatasi->getMessage(dosya->okumaHatasi));
	dosya->Yoket(dosya);
	return 0;
}
```

```
hepsi: derle calistir

derle:
	gcc -I ./include/ -o ./lib/Hata.o -c ./src/Hata.c
	gcc -I ./include/ -o ./lib/DosyaOkumaHatasi.o -c ./src/DosyaOkumaHatasi.c
	gcc -I ./include/ -o ./lib/Dosya.o -c ./src/Dosya.c
	gcc -I ./include/ -o ./bin/Test ./lib/Hata.o ./lib/DosyaOkumaHatasi.o ./lib/Dosya.o ./src/Test.c

calistir:
	./bin/Test
```

Hazırlayan

Dr. Öğr. Üyesi M. Fatih ADAK