Теоретический материал для выполнения ИЗ №3

Гутников С.Е.

Условие ИЗ №3

1) Создать класс, указанный в задании. По возможности использовать assert и исключения для обработки ошибочных ситуаций.

2) В отдельном файле разработать тестовое приложение, использующее класс, указанный в задании. Провести тестирование всех методов и конструкторов с выводом данных и результатов

3) Создать makefile со следующими целями:

- очистка (clean) - удаление промежуточных файлов
- построение (build) – перекомпиляция всех классов и создание исполняемого jar-архива
- запуск (run) – запуск на выполнение jar-архива

Условие ИЗ №3

4) Задание сдается преподавателю в окне консоли (Windows) или в окне терминала (Linux/Unix/MacOS. Кроме исходного кода Java, оценивается также умение студента настроить среду выполнения и его работа с утилитой make.

Утилита Make

Утилита make читает в make-файле, как компоненты программы зависят друг от друга и как их обрабатывать для того, чтобы создать новую версию программы.

Она проверяет время изменения различных компонент, вычисляет минимальный объем рекомпиляции, необходимый для получения новой актуальной версии, и запускает перестройку компонентов программы.

Утилита Маке

Make-файл – это текстовый файл который может содержать следующее:

- комментарии: строки начинающаяся с символа #
- макроопределения: строки вида name=value. Подстановка макроопределения в текст: $(name), переменные среды ОС рассматриваются как предопределённые макроопределения, их можно использовать в тексте, например: $(USERNAME)

Утилита Маке

- явные описания целей, и зависимостей, например:

```
run: build
	java -ea –jar test.jar

z4_2\Matrix.class: z4_2\Matrix.java makefile
	javac –g z4_2\Matrix.java
```

Утилита Маке

- неявные описания целей, и зависимостей. В этих описаниях записывают расширение исходного файла и целевого файла, также используются специальные макросы:

$* -это имя файла без суффиксов

$< -имя файла зависимости

$@ -полное имя цели

например:

```
.java.class:
        javac -g $<
```

Это правило расшифровавается: файл с суффиксом .class получается из файла с таким же именем и суффиксом .java после выполнения указанной команды.

Утилита Маке

Макроопределения и описания целей `make`-файла могут быть очень длинными, их можно задавать в нескольких строках, для продолжения на следующей строке – в конце строки записывается символ `\`

Рассмотрим makefile для решения варианта 2.

Пример

```
# makefile z4_2

.SUFFIXES: .class .java

PACK=z4_2
TARG=test
JC=javac -g
JM=java -ea -jar
JR=jar -cfe

OBJ=$(PACK)\Matrix.class \
    $(PACK)\test.class
```

Пример

```
.java.class:
   $(JC) $<

run: build
   $(JM) $(TARG).jar

build:       $(OBJ)
   $(JR) $(TARG).jar $(PACK).test $(OBJ)

clean:
   for %%f in ($(OBJ)) do del %%f
   del $(TARG).jar
```

Пример

```
$(PACK)\Matrix.class: $(PACK)\Matrix.java makefile

$(PACK)\test.class: $(PACK)\test.java makefile

# eof makefile z4_2
```

Пример

Пример:

```
package z4_2;

/* Matrix.java:
Определить класс Matrix размерности
(n x n). Объявить массив из m объектов.
Написать методы, вычисляющие первую и
вторую нормы матрицы. Определить, какая
из матриц имеет наименьшую первую и
вторую нормы */

import java.util.Random;
import java.util.Date;
```

Пример

```
class Matrix {
    static final int MAX_A = 10;
    int N = 0;
    int [][] mas = null;

    public Matrix() {
        N = 0;
        mas = null;
    }

    public Matrix( int n ) {
        assert( n > 0 );
        Init( n, MAX_A );
    }
```

Пример

```
public Matrix( int n, int max_val )    {
    assert( n > 0 );
    assert( max_val > 0 );
    Init( n, max_val );
}

public int Norm_1() {
    assert( N > 0 );
    int norm = 0;
    for( int i = 0; i < N; i++ ) {
        int temp = 0;
        for( int j = 0; j < N; j++ ) {
            temp += Math.abs( mas[i][j] );
        }
```

Пример

```
            if( temp > norm ) {
                norm = temp;
            }
        }
        return norm;
    }
    public int Norm_2() {
        assert( N > 0 );
        int norm = 0;
        for( int j = 0; j < N; j++ ) {
            int temp = 0;
            for( int i = 0; i < N; i++ ) {
                temp += Math.abs( mas[i][j] );
            }
```

Пример

```
            if ( temp > norm ) {
                norm = temp;
            }
        }
        return norm;
    }

    public void Print() {
        assert( N > 0 );
        for( int i = 0; i < N; i++ ) {
            for( int j = 0; j < N; j++ ) {
                System.out.print( " " + mas[i][j] );
            }
            System.out.println();
```

Пример

```
            }
            System.out.println();
        }
        private void Init( int n, int max_val ) {
            assert( n > 0 );
            assert( max_val > 0 );
            mas = new int [n][n];
            N = n;
            Random rand = new Random(
                (new Date()).getTime() );
            for( int i = 0; i < N; i++ ) {
                for( int j = 0; j < N; j++ ) {
                    mas[i][j] = rand.nextInt( max_val );
}}}}
```

Пример

```
/* test.java: */
package z4_2;
import java.util.Random;

public class test {
    static final int NUM = 10;

    public static void main(String[] args) {
        Matrix [] array = new Matrix[NUM];
        //Random rand = new Random();
        for ( int i = 0; i < NUM; i++ ) {
            System.out.println( i + 1 );
            array[i] = new Matrix(
                /*rand.nextInt(5)+1*/ 2 );
```

Пример

```
        array[i].Print();
    }
    int min = Integer.MAX_VALUE, k = NUM + 1;
    for ( int i = 0; i < NUM; i++ )    {
        int norm = array[i].Norm_1()+
            array[i].Norm_2();
        if ( norm < min ) {
            min = norm;
            k = i + 1;
        }
    }
    System.out.println(
        "matrix #"+ k +" has minimal norm-1 ("+
        array[k - 1].Norm_1()+
```

Пример

```
        ") and norm-2 ("+
        array[k - 1].Norm_2()+")");
for ( int i = 0; i < NUM; i++ ) {
        int norm = array[i].Norm_1();
        if ( norm < min ) {
            min = norm;
            k = i + 1;
        }
}
System.out.println(
        "matrix #" + k +
        " has minimal norm-1 ("+
        min + ")" );
min = Integer.MAX_VALUE;
```

Пример

```
        for ( int i = 0; i < NUM; i++) {
            int norm = array[i].Norm_2();
            if ( norm < min ) {
                min = norm;
                k = i + 1;
            }
        }
        System.out.println( "matrix #"+
            k+ " has minimal norm-2 ("+
            min + ")" );
        System.out.println("---test assert---");
        Matrix m = new Matrix();
        m.Print();
    }
}
```

Дополнительная информация

Смотрите более подробный материал по utilite make в файлах:

- make-1.xps
- make-2.xps