Лабораторная работа № 6

**Обработка исключений.**

**Цель работы:** Формирование навыка обработки ошибок в виде исключений, повышающих надёжность создаваемых программ. Формирование навыка создания проекта по техническим требованиям. Закрепление навыков работы с классами.

**Теория**

Развитый механизм обработки ошибок – это надёжный путь к повышению надёжности программ.На практике программисты часто игнорируют ошибки (например возвращает количество успешно выведенных символов, однако никто не проверяет результат её вызова – это увеличение кода программы). В классическом Си вызов функции тесно связан с обработкой ошибок, код обработки «перемешан» с основным кодом. В С++ механизм обработки исключений позволяет сначала описать нормальное течение вычислений, а затем, в отдельной секции, описать код для решения различных проблем. Таким образом при многократном вызове функции обработка ошибок этой функции производится в одном и том же месте.

Обработка исключений – это системные средства для обработки ошибок времени выполнения.

Инструкции try, сatch, throw образуют взаимосвязанную подсистему

**Общие правила:**

1. throw генерирует (выбрасывает) исключение, которое перехватывается сatch-инструкцией (обработчик ошибок)
2. try-блок определяет отрезок программы, в котором может возникнуть исключение (т.е. где проверяются ошибки)
3. с try-блоком может быть связано несколько сatch-инструкций
4. какой именно сatch- блок будет выполняться определяет типом исключения (встроенный или пользовательский)
5. Инструкция throw exception должна выполняться либо в блоке try, либо в функции вызванной из блока try
6. Если генерируекся throw exception для колорого нет обработчика (сatch-блока), произойдет аварийное завершение программы terminate()

По-умолчанию выполняется abort(), но реакцию можно изменить.

Пример:

void main()

{ cout<<"Begin\n";

try {

cout<<"in try block\n";

throw 99; // генерация int-исключения

cout<<"do not work!!!";

}

// Обработчик ошибок

catch (double k)

{cout<<"catch double exception:"

<< k << "\n" ;

}

catch (int i)

{cout<<"catch int exception:"

<< i << "\n" ;

}

cout<<"The end"<<endl;

system ("pause");

return 0;

}

Результат работы программы :

Begin

in try block

catch int exception: 99

**Перехват исключений классового типа**

Большинство реальных программ генерируют исключения типа класс

**Пример:**

class except

{public:

char msg[80];

except() {\*msg=0;}

except(char\* s) {strcpy(msg,s);}

}; //------- end except

double fdiv (double a, double b)

{if (b)

{ cout<<"+++"<<endl;

return (a/b);}

else throw except("error: divider=0");

}

double fsqrt (double q)

{if (q<0) throw except("error: sqrt");

return sqrt(q);

}

double fasin (double r)

{if (r<-1 || r>1)

throw except("error fasin");

return asin(r);

}

void main()

{ double x1,y1,z1;

cout<<"x1="; cin>>x1;

cout<<"y1="; cin>>y1;

try

{cout<<"z1="<<fdiv (x1,y1)<<endl;

cout<<"z2="<<fsqrt (x1)<<endl;

cout<<"z3="<<fasin (y1)<<endl;

}

catch (except e)

{cout<<e.msg<<endl;

}

сout<<”end”;

}

**Перехват всех исключений**

Если различные исключения обрабатываются одинаково, то используется обработчик исключений вида catch (...)

Следующий пример показываент эту ситуацию. В функции capture () могут генерироваться исключения различного типа, которые обрабатываются одним блоком catch (...)

**Пример:**

void capture (int k)

{try {

switch (k) {

case 1 : throw 1;

case 2: throw 'a';

case 3 : throw 2.8;

}

}

catch (...) // перехват всех исключений

{cout<<"all type capture\n";}

}

void main()

{ cout<<"Begin\n";

capture (1);

capture (2);

capture (3);

cout<<"The end"<<endl;

}

Результат на экране :

Begin

all type capture

all type capture

all type capture

The end

Если в наличии несколько catch – инструкций, и catch (...) последняя из них, то будет перехват «всех остальных»

**Пример:**

void capture (int k)

{try {

switch (k) {

case 1 : throw 1;

case 2: throw 'a';

case 3 : throw 2.8;

case 4 : throw "string";

case 5 : throw except("error!!!"); }

}

catch (int i) {

cout<<"catch int exception:" <<i<<endl;}

catch (char s) {

cout<<"catch char exception:" <<s<<endl;}

catch (except e)

{cout<<e.msg<<endl;

}

catch (...) // перехват всех остальных исключений

{cout<<"all type capture\n";}

}

void main()

{ cout<<"Begin\n";

capture (1);

capture (2);

capture (3);

capture (4);

capture (5);

cout<<"The end"<<endl;

}

Результат на экране :

Begin

catch int exception: 1

catch int exception: а

all type capture

all type capture

error!!!

The end

**Общие правила и требования**

1. При оформлении ввода-вывода данных информация на экране должна быть отформатирована:

* на экран выводится тема задания (кратко);
* ввод данных и результат вычислений выводить с комментариями;
* выделять области ввода и вывода информации с помощью строк-разделителей.

1. В данной лабораторной работе вы учитесь создавать проект по техническому заданию (постановка задачи).
2. В задании указан набор обязательных элементов проекта, но при реализации взаимодействия классов от вас потребуется добавить элементы:

* необходимые для работы (счетчики, размеры массивов и т.п.);
* не обязательные, но упрощающие реализацию алгоритмов обработки данных (для промежуточных данных, для хранения общих данных и т.п.)
* вы также можете усложнить иерархию и расширить интерфейсы в рамках поставленной задачи (это добавит вам баллы при защите работы).

1. Общие требования к проекту:

* Данные во всех классах должны быть защищенные (private или protected).
* Хотя бы в одном классе должен быть внедренный объект (в.о.) другого класса
* Обработка любых ошибок должна реализовываться в виде исключений
* Первоначальную информацию в базу нужно загружать с диска, имя файла и путь задать в программе, обработать ошибку открытия файла.
* Данные, вносимые по ходу работы программы должны вводиться с клавиатуры, предусмотреть контроль корректности ввода и обработку ошибки. Например:
* контроль диапазона значений: 1-12(для месяца), 0-24(для часа), 1-8(число отработанных часов в день);
* запрет деления на ноль и т.п.

1. Первый этап работы над проектом – определение структуры программы: классы и их иерархическое взаимодействие (в иерархии должно быть как минимум два уровня). Следует определить, какой класс будет базовым, а какой (или какие) – производным, в каких классах расположить встроенные объекты (вы можете увеличить число встроенных объектов, если это целесообразно).
2. В задании ничего не говорится о конструкторах и деструкторах, но для корректной работы они необходимы. Второй этап работы над проектом: определиться с конструкторами и деструкторами: какого типа и сколько их необходимо.
3. Интерфейс должен быть реализован как с помощью методов класса, так и с помощью дружественных функций. В проекте должны быть как методы, так и дружественные функции.
4. Интерфейс необходимо реализовать в виде меню команд с возможностью многократного их выполнения.
5. В задании ничего не говориться о выводе информации на экран, но от решения этого вопроса зависит качество проекта в целом. В целом вывод можно разделить на три категории:

* информация выводится по умолчанию при запуске программы;
* информация выводится по запросу пользователя (пункт меню);
* вывод информации – это результат выполнения какой-либо операции.

В вашем проекте должны быть все три категории вывода. Должен быть пункт меню «распечатать всю информацию».

Третий этап работы над проектом: проектирование ввода-вывода и стиля представления информации на экране. Информация должна «хорошо читаться», красиво выглядеть и быть удобной для восприятия. Ориентируйтесь на стандартные решения, которые нам хорошо известны по работе в Интернете.

Третий этап разработки проекта лучше выполнять в последнюю очередь. Сначала добейтесь, чтобы программа работала и выполняла все заданные функции. Стиль вывода на этом этапе не важен, он нужен только для отладки программы. Затем следует спроектировать и реализовать ввод-вывод для пользователя.

1. Во многих заданиях есть позиции-даты:

«по плану» означает контрольную дату выполнения (как правило задается при создании объекта);

«по факту» означает фактическую дату выполнения (задается в процессе работы).

1. Оценка лабораторной работы (всего 15 баллов) складывается из трех частей:

* Проект базы данных (6 баллов)
* Интерфейс пользователя (4 балла)
* Выполнение и защита работы (5 баллов)

|  |  |
| --- | --- |
| Вариант задания | Задание |
| 1, 15 | Постановка задачи: создать базу данных «Табель (предприятие)» для учета отработанного времени в течении года и расчета заработной платы сотрудника. С диска загрузить информацию о нескольких сотрудниках и их табели. Обязательные элементы-данные:   1. Сотрудник: имя, адрес; год рождения; 2. строка табеля : массив (int) из 31 элемента (число дней в месяце), каждый элемент число отработанных часов в день   Обязательные элементы-функции:   1. Добавить нового сотрудника (информация о сотруднике + строка табеля за 1-ый месяц) 2. Записать информацию в табель (число отработанных часов за определенный месяц и день) 3. Вывести на экран все табели заданного сотрудника. Контроль ошибки («такого сотрудника нет») |
| 2, 16 | Постановка задачи: создать базу данных «Отдел снабжения (предприятие)» для учета выполнения заявок на закупки. С диска загрузить информацию об отделе снабжения(сотрудники и заявки).  Обязательные элементы-данные:   1. Заявка: название позиции, количество, цена, дата выполнения (по факту), исполнитель (сотрудник). 2. Список сотрудников отдела снабжения (ФИО, должность, оклад)   Обязательные элементы-функции:   1. Добавить сотрудника 2. Распечатать все выполненные заявки (с заполненной датой «по факту») 3. Определить список заявок для заданного исполнителя. Контроль ошибки («такого исполнителя нет») |
| 3, 17 | Постановка задачи: создать базу данных «Отдел кадров (предприятие)» для учета сотрудников предприятия и их зарплаты. С диска загрузить информацию о нескольких сотрудниках различных отделов предприятия.  Обязательные элементы-данные:   1. Сотрудник: имя, год рождения; отдел; дата поступления на работу 2. Зарплата : массив (double) из 12 элементов (число месяцев в году). Для каждого сотрудника хранится информация о зарплате за несколько лет.   Обязательные элементы-функции:   1. Добавить нового сотрудника (информация о сотруднике + массив для зарплаты за текущий год) 2. Добавить год (новый массив из 12 элементов для учета зарплаты) для заданного сотрудника. Контроль ошибки («такого сотрудника нет») 3. Записать информацию о зарплате сотрудника за определенный год и месяц 4. Рассчитать годовую зарплату заданного сотрудника за определенный год. Контроль ошибки («некорректное значение года», диапазон значений - от года поступления на работу до текущего года) |
| 4, 18 | Постановка задачи: создать базу данных для расчета сырья по калькуляции.  С диска загрузить несколько изделий и калькуляций.  Пояснение: для производства (например, пищевого) требуется расчет сырья (для отпуска со склада). Каждое изделие имеет свой список позиций сырья и количество этого сырья – это калькуляция (например изд.1: мука 5.5, масло 0.5, сахар 1.5 и т.п.). Для булок и пирожных калькуляция составлена на 100 шт, для тортов – на 10 шт. Заказ – это перечень и количество изделий, которые нужно изготовить. Расчет заказа – это определение суммарного количества сырья по каждой позиции в калькуляциях для выполнения всего заказа. Например, в заказе 3 позиции: Булочка «свердловская» 3шт., пирожное «Эклер» 5 шт., торт «киевский» 10 шт Нужно рассчитать количество сырья по каждой позиции калькуляции на весь заказ, например: Мука 10.45 кг, масло 1,44 кг и т.д.  Обязательные элементы-данные:   1. Изделие: код, название, тип (булка, пирожное, торт) 2. Калькуляция на изделие (список): изделие(код), сырье(название), количество сырья 3. Заказ на производство (список): номер заказа, изделие(код), количество 4. Список сырья (результат расчета): номер заказа, сырье (название), количество   Обязательные элементы-функции:   1. Создать заказ (список и количество изделий). Контроль ошибки («такого изделия нет») 2. рассчитать заказ (создать список сырья для отпуска со склада). |
| 5, 19 | Постановка задачи: создать базу данных «Плановый отдел (предприятие)» для учета тем предприятия и сроков их выполнения. С диска загрузить информацию о нескольких темах.  Обязательные элементы-данные:   1. Тема (разработка): название, количество этапов 2. График (этапы сдачи): номер этапа, дата сдачи этапа (по плану), дата сдачи этапа (по факту)   Обязательные элементы-функции:   1. Добавить новую тему (определяются все плановые сроки по этапам) 2. Записать информацию о выполненном этапе заданной темы 3. Определить темы с нарушением графика сдачи для заданной даты   Контроль ошибки («некорректная дата») |

|  |  |
| --- | --- |
| 6, 20 | Постановка задачи: создать базу данных «Поставщики товара» для расчета с поставщиками товара. С диска загрузить информацию о нескольких поставщиках. Обязательные элементы-данные:   1. Товар: наименование, цена 2. Поставщик: имя, код, общая сумма за товар, оплата товара(сумма) 3. Приходная накладная (список): поставщик(код), товар, количество   Обязательные элементы-функции:   1. Создание приходной накладной (список) и расчет общей стоимости товара: добавление стоимости товара по накладной к общей сумме за товар. 2. Оплата поставщику за товар. Контроль ошибок: «такого поставщика нет». 3. Отобразить весь список товаров от данного поставщика (по всем накладным) 4. Отобразить поставщиков с положительным сальдо (оплата за товар превышает стоимость товара) |
| 7, 21 | Постановка задачи: создать базу данных «Отдел (предприятие)» для учета сотрудников и тем отдела. С диска загрузить информацию об отделе (сотрудники и темы). Обязательные элементы-данные:   1. Отдел: название, начальник отдела, количество сотрудников. 2. Сотрудник: ФИО, должность (разработчик, инженер, лаборант), в какой теме участвует. 3. Список сотрудников 4. Список тем (название, номер, срок сдачи)   Обязательные элементы-функции:   1. Добавить сотрудника 2. Распечатать информацию по должностям (по всем лаборантам и т.д.) Контроль ошибки («такой должности нет») 3. Определить темы со сроком в заданном месяце. |
| 8, 22 | Постановка задачи: создать базу данных «Торговая точка, продажи» для учета товаров. С диска загрузить информацию о торговой точке: товар на складе и расходные накладные. Обязательные элементы-данные:   1. Товар: наименование, цена 2. Товар на складе (список): товар, количество 3. Расходная накладная:  * Шапка накладной : номер, дата * список товаров: товар, количество   Обязательные элементы-функции:   1. Создание расходной накладной (шапка и список ) 2. Списание товара со склада по накладной: поиск товара согласно накладной и уменьшение его количества на складе.   Контроль ошибки («такого товара нет» , «товара недостаточно»)   1. Определение списка продаж и суммы продаж за определенный день 2. Определение списка определенного товара на складе |

|  |  |
| --- | --- |
| 9, 23 | Постановка задачи: создать базу данных «Табель (предприятие)» для учета отработанного времени в течении года и расчета заработной платы сотрудника. С диска загрузить информацию о нескольких сотрудниках.  Обязательные элементы-данные:   1. Сотрудник : имя, год рождения; разряд/оклад, руб. в час (double) 2. строка табеля : массив (int) из 31 элемента (число дней в месяце), каждый элемент число отработанных часов в день   Обязательные элементы-функции:   1. Добавить новую строку в табель (начать новый месяц) для заданного сотрудника. Контроль ошибки («такого сотрудника нет», «некорректное задание месяца») 2. Записать информацию в табель (число отработанных часов за определенный месяц и день) 3. Вычислить зарплату сотрудника за определенный месяц (суммарное число отработанных часов \* разряд/оклад) |
| 10, 24 | Постановка задачи: создать базу данных «Отдел снабжения (предприятие)» для учета выполнения заявок предприятия. С диска загрузить информацию об отделе (сотрудники и заявки).  Обязательные элементы-данные:   1. Заявка: номер отдела, название позиции, количество, цена,   дата выполнения (по факту), исполнитель (сотрудник).   1. Список сотрудников отдела снабжения (ФИО, должность, оклад)   Обязательные элементы-функции:   1. Добавить заявку 2. Отметить выполнение заявки (заполнить дату по факту). Контроль ошибки («некорректная дата») 3. Распечатать все невыполненные заявки (заявки без даты «по факту») 4. Определить выполненные заявки за определенный месяц |
| 11, 25 | Постановка задачи: создать базу данных «калькуляции на изделия» для работы с калькуляциями. С диска загрузить несколько изделий и калькуляций.  Пояснение: для производства (например, пищевого) требуется расчет сырья (для отпуска со склада). Каждое изделие имеет свой список позиций сырья и количество этого сырья – это калькуляция (например «булка Московская»: мука 5.5, масло 0.5, сахар 1.5 и т.п.). Для булок и пирожных калькуляция составлена на 100 шт, для тортов – на 10 шт.  Обязательные элементы-данные:   1. Изделие: код, название, тип (булка, пирожное, торт), срок годности 2. Калькуляция на изделие (список): изделие (код), сырье(название), количество сырья   Обязательные элементы-функции:   1. Добавить изделие и калькуляцию 2. Удалить изделие и калькуляцию. Контроль ошибки («такого изделия нет») 3. Распечатать калькуляцию на заданное изделие 4. Распечатать все изделия |
| 12, 26 | Постановка задачи: создать базу данных «Отдел (предприятие)» для учета сотрудников и тем отдела. С диска загрузить информацию об отделе (сотрудники и темы). Обязательные элементы-данные:   1. Отдел: название, начальник отдела, количество сотрудников. 2. Сотрудник: ФИО, должность (разработчик, инженер, лаборант), в какой теме участвует. 3. Список сотрудников 4. Список тем (название, номер, дата сдачи)   Обязательные элементы-функции:   1. Добавить тему. Контроль ошибки («некорректное задание даты») 2. Распечатать информацию по заданной теме (сотрудники, занятые в теме). Контроль ошибки («такой темы нет») 3. Определить темы со сроком в заданном году. |
| 13, 27 | Постановка задачи: создать базу данных «Плановый отдел (предприятие)» для учета тем предприятия и сроков их выполнения. С диска загрузить информацию о нескольких темах.  Обязательные элементы-данные:   1. Тема: название темы, отдел в котором выполняется, разработчик, дата сдачи темы (по плану), дата сдача темы (по факту).   Обязательные элементы-функции:   1. Добавить новую тему (определяется все, кроме дата сдача (по факту). 2. Записать информацию о выполнении темы (дата по факту) 3. Определить все темы заданного разработчика. Контроль ошибки («такого разработчика нет») 4. Определить все темы заданного отдела. Контроль ошибки («такого отдела нет») |
| 14, 28 | Постановка задачи: создать базу данных «Торговая точка, приход» для учета товаров. С диска загрузить информацию о торговой точке: товар на складе и приходные накладные.  Обязательные элементы-данные:   1. Товар: наименование, цена 2. Товар на складе (список): товар, количество 3. Приходная накладная:  * Шапка накладной : номер, дата, поставщик * список товаров: товар, количество   Обязательные элементы-функции:   1. создание новой накладной (шапка и список) 2. Приход товара по накладной: добавление позиции к складу (если нет товара, указанного в накладной) и увеличение количества для позиции существующего товара, указанного в накладной. 3. Распечатка всего склада (товар на складе) 4. Определение списка определенного товара на складе. Контроль ошибки («такого товара нет») |