Курс «Основы программирования»

Власенко Олег Федосович
SimbirSoft

Практика 1.

БС1. IF & DO WHILE – блок-схемы и трассировка

СП1. Сквозной проект – этап 1

СП - Сквозной проект

Что это такое? СТАРТ!

Что такое «Сквозной проект» и зачем он вам?

Чтобы стать ИТ специалистом, необходимо не только изучить основы основ, но и получить опыт работы на достаточно больших проектах. Поэтому в рамках курса «Основ программирования» кроме простых лабораторных работ (и усложненных задач к этим лабораторным работам) есть еще Сквозной проект, в котором в течение семестра каждый студент может получить опыт длительной разработки достаточно объемного проекта.

Сквозной проект выполняется СТРОГО ПО ЭТАПАМ.
Он может быть сделан быстро досрочно – и он будет зачтен, в случае если авторство будет бесспорно.
Но в случае сильного запаздывания по этапам этот проект (даже сделанный! И даже при бесспорном авторстве!!!!!) не будет засчитан!

В рамках Сквозного проекта каждый студент будет делать свой собственный (уникальный!!!) вариант **ИГРЫ**.

Какой вариант игры уже сделан для демонстрации?

Название игры: «Жизнь ёжика»

Объекты (Персонажи и предметы): Ёжик, грибы (его еда), яблоки (его лекарства), лиса (его враг), нож (его оружие)

Сценарий основной: Ёжик должен собрать все яблоки на уровне – тогда он победил.

Сценарии другие: Есть лиса - враг ёжика. Если он встретился с ней больше раз, чем следовало – игра проиграна. После встреч с лисой ёжик может подлечиться, если будет есть яблоки. Если ёжик обзаведется ножом, то при встрече с лисой не поздоровиться уже лисе.

Управление: Ёжик передвигается при нажатии клавиш – влево/вправо, вверх/вниз. Лиса перемещается автоматически – случайно.

Какой вариант игры должен выбрать и реализовать студент?

Вам предстоит в самом начале выбрать идею для своей **игры**.

Эта идея должна быть
А) отличной от идей всех ваших одногруппников
Б) Всё, что будет сделано в игре, вы либо уже знаете как сделать на момент начала работы над игрой, либо вы изучите все необходимое в рамках курса «Основы программирования»
В) Всю игру вы должны сделать в достаточной степени самостоятельно. Допустимо консультироваться или заимствовать код, но

1) любая строчка кода вами понята и может быть легко (и быстро) объяснена
2) более 50% строк кода написано вами индивидуально без какой-либо помощи и заимствований.

Если вы придумываете свой собственный вариант, то вы обязаны его согласовать с вашим преподавателем по лаб работам. Если он его одобрит – лишь после этого вы можете приступать к реализации.

В случае наличия списка с вариантами, вы можете выбрать вариант из предоставленного списка игр. В этом случае гарантируется, что вы в состоянии будет реализовать эту игру в рамках этого семестра, и что все нужные для проекта знания включены в темы лекций и лаб работ.

Какие требования к Сквозному проекту?

1) Проект выполняется на Си в среде Microsoft Visual Studio
2) Проект выполняется персонально. Кода, написанного лично, должно быть не менее 50%
3) Каждая строка позаимствованного кода (в том числе и сгенерированного VS, созданного в кооперации с консультантами, найденного в интернете и т.п.) должна быть понятна сдающему. (Понятность проверяется как опросом, так и заданием что-то в этом коде изменить)
4) Сквозной проект является ДОПОЛНИТЕЛЬНЫМ проектом. В случае его отсутствия у студента, студент имеет возможность получить положительную оценку за дисциплину.
5) Сквозной проект выполняется либо четко по этапам, либо с опережением этапов. Если студент отстает от графика и группы, то проект ему не засчитывается. Совсем не засчитывается ☹ (это делается как для того, чтобы студенты привыкали к четкому графику, так и для уменьшения возможностей списывания)

Варианты игры

Подход 1. Изучите реализованную игру «Жизнь ёжика». Подумайте, какие герои могут быть в вашей версии игры? Какой сценарий будет у вас? Например, можно сделать игру (по аналогии с «Жизнью ёжика») где «Девочка» собирает «Цветы», чтобы сплести «Венок». Объект «Крапива» представляет для нее определенную угрозу. А если она догонит «Бабочку», то получает бонус.
После того как придумаете игру, опишите все объекты и сценарии в виде текста. И согласуйте вариант вашей игры с преподавателем ведущим лаб работы.

Подход 2. Выберите готовый вариант из списка
(СПИСОК В РАЗРАБОТКЕ. В настоящий момент он недоступен для выбора вариантов!)

Вопрос – а когда заниматься Сквозным проектом? И когда показывать этапы преподавателю?

Когда им заниматься?

Сквозной проект – не обязательная часть программы.
Это не требование – это бонус вам!

Если хотите прокачать себя в разработке ПО и стать ИТ профессионалом – лучшее, что можно для этого сделать – это наработать опыт, написав 1000-1500 строк кода, и потратив на их написание 60+ часов в течение 4 месяцев.

Это время - время вашей самостоятельной работы.

Когда сдавать?

Периодически (1 раз в 3 недели) будет специальное занятие, на котором вы обязаны продемонстрировать ваши наработки преподавателю, ведущему лаб работы.
Если у вас наработки будут раньше – можете продемонстрировать их раньше – на очередных практических занятиях.

А если будут вопросы?

Вопросы можно задавать как на лекциях, так и на практических занятиях.

Сквозной проект – этап 1 (СП1)

Выбор варианта игры.
Отрисовка всех объектов.
(Статическая картинка).
Презентация итогов этапа 1 преподавателю.

Задачи ЭТАПА 1

Задача 1. Выбор темы игры
Задача 2. Утверждение темы игры
Задача 3. Рисуем все персонажи и предметы в виде картинки (в компьютере или на бумаге)
Задача 4. Создать приложение, содержащее все объекты из игры
Задача 5. Презентация преподавателю получившегося приложения на занятии

Задача 1. Выбор темы игры

Вам необходимо выбрать или придумать тему для игры и оформить ее в виде текста. В описание вашего варианта рекомендуется придерживаться следующей структуры.

Название игры: *«Жизнь ёжика»*

Объекты (Персонажи и предметы): *Ёжик, грибы (его еда), яблоки (его лекарства), лиса (его враг), нож (его оружие)*

Сценарий основной: *Ёжик должен собрать все яблоки на этом уровне – тогда он переходит на следующий. Если он прошел все уровни – он победил.*

Сценарии другие: *Есть лиса - враг ёжика. Если он встретился с ней больше раз, чем следовало – игра проиграна. После встреч с лисой ёжик может подлечиться, если будет есть яблоки. Если ёжик обзаведется ножом, то при встрече с лисой не поздороваться уже лисе.*

Управление: *Ёжик передвигается при нажатии клавиш – влево/вправо, вверх/вниз. Лиса перемещается автоматически – в сторону к ежику или случайно.*

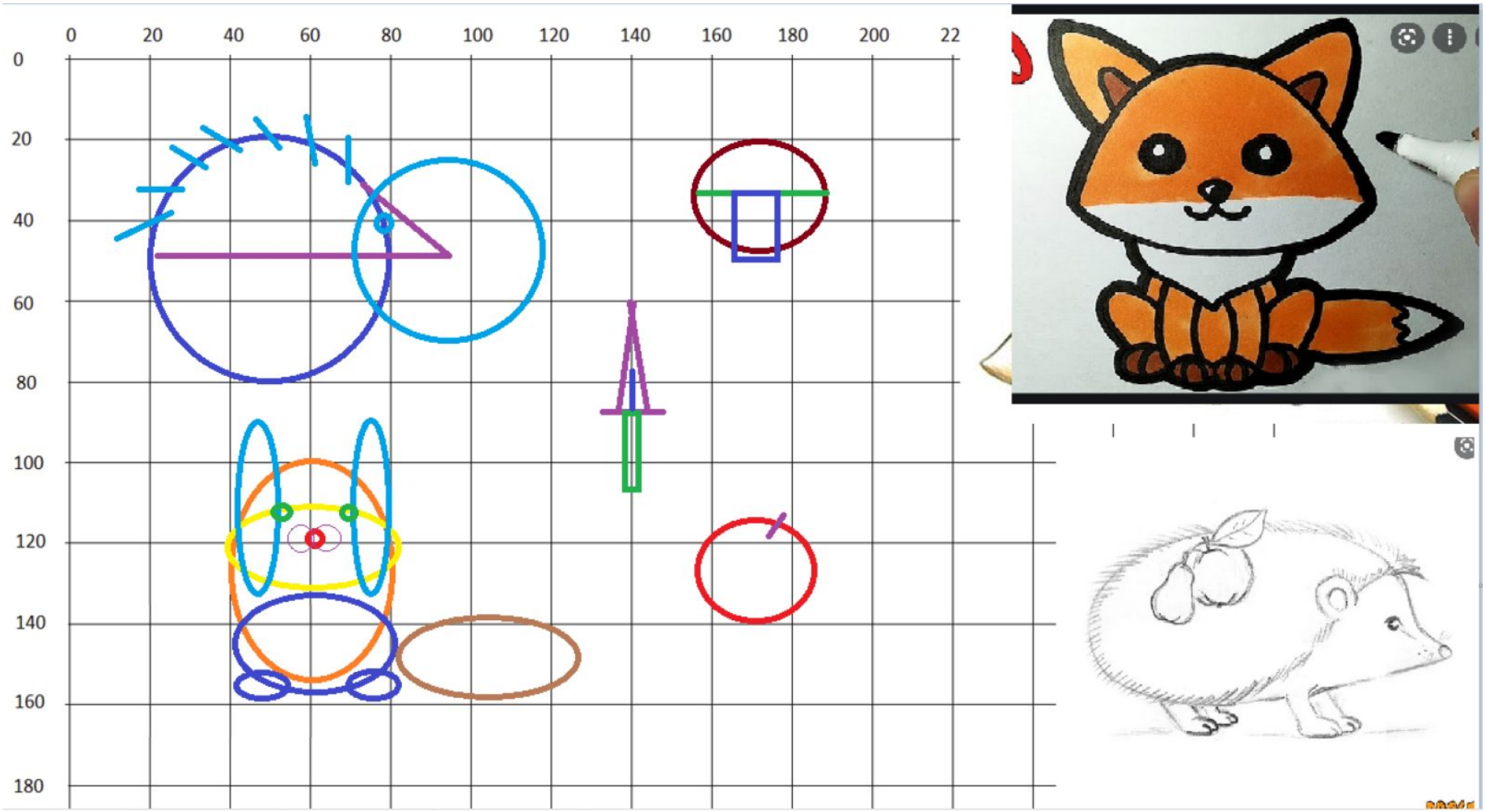
Задача 2. Утверждение темы игры

После выбора темы вам необходимо утвердить её у вашего преподавателя, ведущего лабораторные работы.

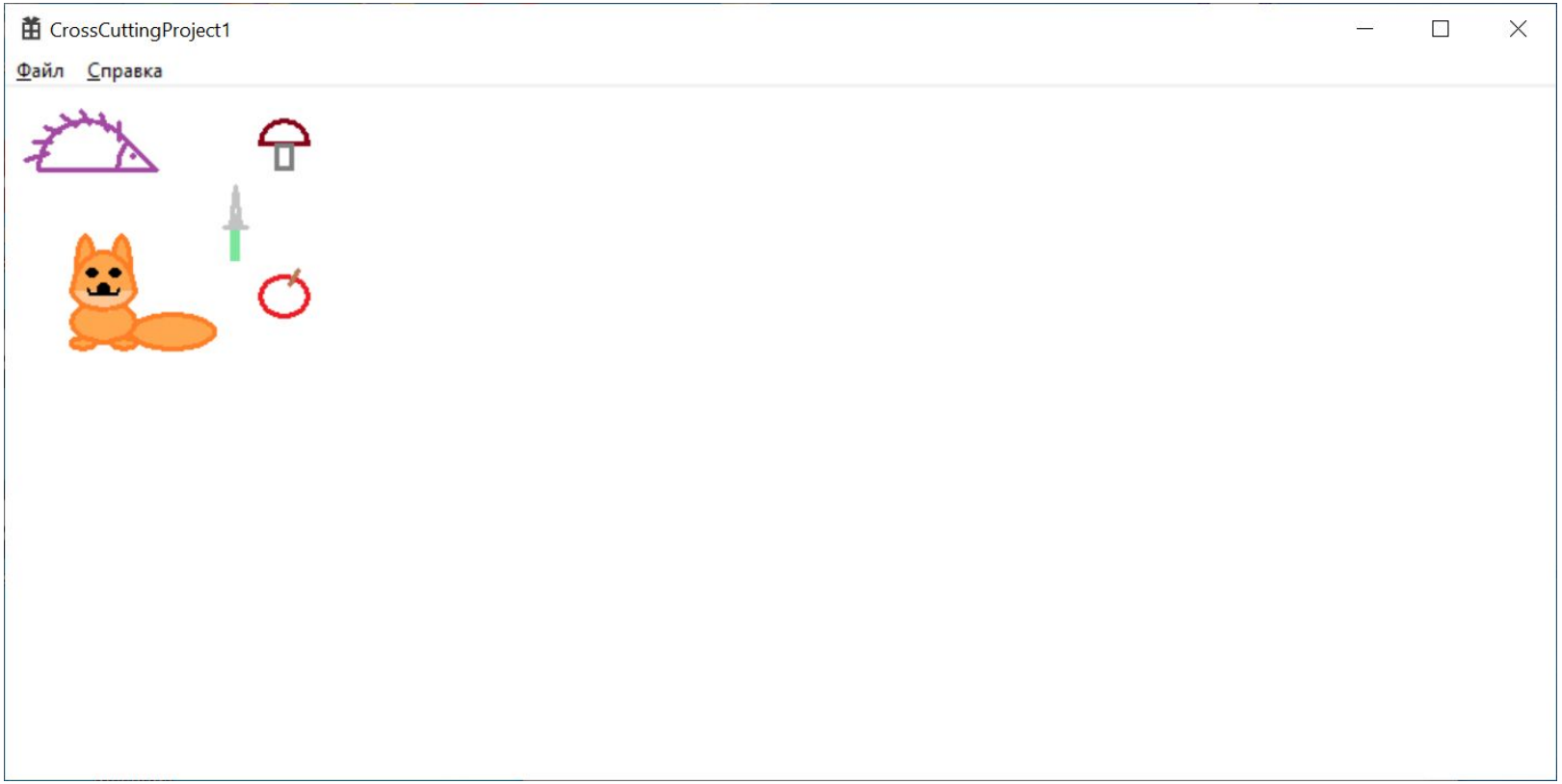
До утверждения темы не рекомендуется начинать кодирование.

Сразу после утверждения вы можете выполнять Задачу 3 и далее.

Задача 3. Рисуем все персонажи и предметы в виде картинки (в компьютере или на бумаге)



Задача 4. Создать приложение, содержащее все объекты из игры



Задача 5. Презентация преподавателю получившегося приложения на занятии

Вам необходимо показывать ваши наработки по игре преподавателю после выполнения КАЖДОГО ЭТАПА проекта!

Если вы выполнили все этапы, но показали всё только в конце, то проект вам не засчитывается. ДАЖЕ ЕСЛИ ВЫ ДОКАЗАЛИ полное свое авторство – увы ☹️

ИТОГО по этапу 1

1. Вы выбрали вариант игры и согласовали его с преподавателем
2. Описали игру – объекты и сценарии
3. Создали картинку (на бумаге или в компьютере) с объектами
4. Создали программу отрисовывающую все объекты
5. Продемонстрировали программу и картинку преподавателю

У вас есть всё, чтобы идти дальше и реализовать сценарии игры!

БС1 – блок-схемы и трассировка

IF & DO WHILE

Зачем вообще блок-схемы и трассировка

Чтобы программист смог «заставить» компьютер выполнять написанную им программу, он обязан четко понимать как работают команды, которые он отдает компьютеру – как каждая по отдельности, так и все вместе.

Только лишь если программист понимает это, он сможет составить корректную программу.

Естественно, для успешного решения бизнес-задачи при помощи ИТ, нужно много чего другого. Нужно
А) понять, как именно выполняется задача (бизнес-аналитика)
Б) нужно «вытащить» алгоритм или его разработать
В) нужно подобрать необходимые структуры данных
Г) нужно создать код программы, в котором будет реализован алгоритм, и будут использованы структуры данных
Д) нужно отладить эту программу, и довести её до работоспособного состояния
Е) нужно протестировать программу
Ж) нужно передать её заказчику и обучить пользователей
З) нужно обеспечить поддержку пользователей
И сделать еще много другого!

Но всё это БЕССМЫСЛЕННО, если «программист» не в курсе, как работает компьютер, как компьютер исполняет написанную программу!
Блок-схемы и трассировка нужны для того, чтобы вы разобрались как именно компьютер выполняет тот код, который вы ему отдаете на выполнение!

Что будем делать?

Сегодня (до конца пары) я разберу несколько задач.
На ближайшем практическом занятии, у вас будет самостоятельная работа, где вы по вариантам решите аналогичные задачи, и сдадите их вашим преподавателям.

Каждая из задач представляет собой
А) код программы на СИ (корректной программы)
Б) конкретных входных данных

Вам нужно в каждой задаче сделать
А) Нарисовать **на бумаге** блок-схему программы.
Б) Выполнить **на бумаге** трассировку этой программы.

Задача 1А – на разбор

```
// Задача 1. Вариант А
#include <stdio.h>
void main() {
     int a, b, c, d, e;

     scanf_s("%d%d%d", &a, &b, &c);
     d = 0;
     e = 0;

     // если a делится на 2 без остатка
     if (a % 2 == 0) {
          d++;
          e += a;
     }
     // если b делится на 2 без остатка
     if (b % 2 == 0) {
          d++;
          e += b;
     }
     // если c делится на 2 без остатка
     if (c % 2 == 0) {
          d++;
          e += c;
     }

     printf("%d %d", d, e);
}
```

Тест 1. Введите
1 2 3

Тест 2. Введите
2 4 6

Задача 1В – на самостоятельную отработку

```
// Задача 1. Вариант В
#include <stdio.h>
void main() {
     int a, b, d, e;

     scanf_s("%d%d", &a, &b);
     d = 0;
     e = 1;

     // если a делится на 3 без остатка
     if (a % 3 == 0) {
          d++;
          e *= a;
     }
     // если b делится на 3 без остатка
     if (b % 3 == 0) {
          d++;
          e *= b;
     }

     if (d == 0) {
          e = 0;
     }

     printf("%d %d", d, e);
}
```

Тест 1. Введите
1 2

Тест 2. Введите
3 6

Задача 2A – на разбор

```c
// Задача 2. Вариант А
#include <stdio.h>
void main() {
    int a, b;

    scanf_s("%d", &a);
    b = 1;
    do {
        printf("%d ", b);
        b++;
    } while (b <= a);

    do {
        b--;
        printf("%d ", b);
    } while (b > 1);
}
```

Тест 1. Введите
3

Задача 2В – на самостоятельную отработку

```
// Задача 2. Вариант В
#include <stdio.h>
void main() {
    int a, b;

    scanf_s("%d", &a);
    b = a;
    do {
        printf("%d ", b);
        b--;
    } while (b >= 1);

    do {
        b++;
        printf("%d ", b);
    } while (b < a);
}
```

Тест 1. Введите
3

Задача 3А – на разбор

```
// Задача 3. Вариант А
#include <stdio.h>
void main() {
     int a, b, c;
     int m2, m3;

     scanf_s("%d%d", &a, &b);

     m2 = 0;
     m3 = 0;

     c = a;
     do {
          if (c % 2 == 0) {
               m2++;
          }
          if (c % 3 == 0) {
               m3++;
          }
          c++;
     } while (c <= b);

     printf("%d %d ", m2, m3);
}
```

Тест 1. Введите
3 6

Задача 3В – на самостоятельную отработку

```
// Задача 3. Вариант В
#include <stdio.h>
void main() {
    int n, d;

    scanf_s("%d", &n);
    printf("%d = 1 ", n);
    d = 2;
    do {
        if (n % d == 0) {
            printf("* %d ", d);
            n = n / d;
        }
        else {
            d++;
        }
    } while (n > 1);
}
```

Тест 1. Введите
30

Тест 2. Введите
300

ИТОГО по практике 1

1. Узнали что такое СП и что нужно делать в СП1
2. Узнали что будет на ближайшей практике - самостоятельная по блок-схемам и трассировке
3. Вспомнили, как рисуются блок-схемы для развилок и циклов DO WHILE
4. Разобрались, как выполняется ручная трассировка на бумаге.