Список использованной литературы

1. Толковый словарь по искусственному интеллекту - http://www.raai.org/library/tolk/aivoc.html#L208
2. Программирование искусственного интеллекта в приложениях / М. Тим Джонс ; Пер. с англ. Осипова А. И. – М.:ДМК Пресс, 2004. – 312с : ил.
3. Документация Django – документация Django 1.9 - https://djbook.ru/rel1.9/
4. Прохоренок Н.А. Python 3 и PyQt. Разработка приложений. – СПб.: БХВ-Петербург, 2012. – 704 с.
5. Лутц М. Программирование на Python, том II, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 992 с
6. Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с
7. PostgreSQL: Документация: 9.6: 1. - https://postgrespro.ru/docs/postgresql/9.6/intro-whatis.html

ГЕНЕРАЦИЯ ЛАБИРИНТОВ ДЛЯ ДЕТЕЙ МЛАДШЕГО ШКОЛЬНОГО И ДОШКОЛЬНОГО ВОЗРАСТА

Автор: Кузина Ирина Александровна, студентка 3 курса филиала «Протвино» государственного университета «Дубна»

Научный руководитель: кандидат технических наук, доцент кафедры филиала «Протвино» государственного университета «Дубна» Кульман Татьяна Николаевна.

Аннотация.

Данное приложение создано в среде разработки программного обеспечения Microsoft Visual Studio 2015. Функциональная часть реализована на языке программирования C++ при помощи стандартной библиотеки шаблонов STL с использованием контейнерного класса stack. Для создания интерфейса также использовалась среда разработки Microsoft Visual Studio 2015, но вся реализация производилась на языке программирования C#.

В данной статье описывается ход создания проекта по генерации лабиринтов для детей младшего школьного и дошкольного возраста. Данная тема представляет интерес с различных точек зрения. Во-первых, это развитие мышления посредством углубленного изучения генерирования лабиринтов и учёта всех возможных условий для его создания. Во-вторых, обучение работе с различными алгоритмами для генерирования лабиринтов, прохода по ним с целью поиска оптимального пути. И в-третьих, приобретение навыков работы с различными языками программирования, умение переносить проект с одного языка программирования на другой, а также создание пользовательского интерфейсов под цели проекта.

Актуальность данного проекта обуславливаются её практической значимостью. Проект может быть полезен для развития логического мышления у детей младшего школьного и дошкольного возраста, ввиду того, что есть возможность генерировать лабиринты от самых простых и маленьких до более сложных и больших.

Если основываться на мнениях психологов, то также можно утверждать, что лабиринты помогают в развитие внимания у ребёнка, развивают моторику ребёнка, а также усидчивость и навыки достижения целей посредством обхода возникающих перед ребёнком препятствий.

Имеется также проблема, заключающаяся в том, что, если лабиринт будет сгенерирован неверно, то есть будет очень запутанным, или вовсе не будет иметь правильного решения, это может негативно отразиться на уровне притязаний (стремление к достижению целей той степени сложности, на которую человек считает себя способным) ребёнка. Потому важной задачей является генерирование лабиринтов, имеющих как минимум одно правильное решение.

Цель данной работы – создание приложения, позволяющего генерировать лабиринты разных размеров, проходить по ним в поисках оптимального пути, создание таблицы учёта путей и их веса после прохождения пользователем по сгенерированным лабиринтам.

При выполнение данной исследовательской работы были поставлены и решены следующие задачи:

- выбор алгоритма для генерирования лабиринтов разных размеров;
- выбор алгоритмов для поиска оптимального пути;
- выбор инструментов для разработки функциональной части приложения;
- выбор инструментов для разработки пользовательского интерфейса с внедрением функциональной части приложения;
- разработка алгоритма для генерирования лабиринтов и его реализация в виде программного кода;
- разработка алгоритма для поиска оптимального пути и его реализация в виде программного кода;
- разработка пользовательского интерфейса для обеспечения необходимой наглядности и функциональности.

Перед реализацией генерирования лабиринтов и выбора алгоритма стоит задача постановки условий, при которых лабиринт может быть сгенерирован. Для создания лабиринта как минимум требуется матрица размерами (4×4), при этом размерность массива может достигать 100 ячеек, и даже более, то есть размерность массива будет ограничиваться размерами (n×m). Стенками лабиринта будут служить следующие строки и столбцы матрицы:

- строки:
    1. от (0,0) до (0,n);
    2. от (m,0) до (m,n);
- столбцы:
    1. от (0,0) до (m,0);
    2. от (0,n) до (m,n).

При этом как минимум на двух стенках лабиринта должно быть две «пустые» ячейки, которые примем за вход в лабиринт и выход из него. Под пустыми ячейками будем понимать возможный путь, по которому можно «пройти». Следует обратить внимание на то, что проход по лабиринту возможен только по следующим направлениям: направо, налево, вверх, вниз. Ввиду этого следует учесть угловые части лабиринта, а именно (0,0), (0,n), (m,0) и (m,n) – данные ячейки не могут содержать в себе вход в лабиринт, либо выход из него. Также, при генерирование больших лабиринтов, на одной стенке лабиринта может быть более одной пустой ячейки, являющейся либо входом, либо выходом, но данные ячейки не должны стыковаться между собой, то есть между ними как минимум должна быть одна «заполненная» ячейка. Под «заполненной» ячейкой будем понимать некоторый блок-стенку, являющуюся ограждением. Также, чтобы лабиринт был правильным, в нём должен быть как минимум один возможный путь. Это один из возможных путей решения данной задачи.

Далее более детально рассмотрим один из используемых мной алгоритмов, который также можно использовать и для прохода по уже сгенерированному лабиринту с целью поиска оптимального пути из нескольких возможных или для поиска одного возможного пути. При помощи алгоритма, основанного на бэктрекинге, можно создавать лабиринты без циклов, но имеющий лишь один единственный путь, который пролегает между двумя точками. Данный алгоритм довольно прост в своей реализации, хотя и не является самым быстрым.

При использование данного алгоритма будем учитывать то, что изначально имеем стенки с каждой из четырёх сторон у каждой ячейки. Эти стенки будут отделять каждую ячейку от других соседних ячеек. Изначально устанавливаем начальную ячейку, делаем её текущей и отмечаем как уже посещённую. Далее, пока имеем не посещённые ячейки, можем пойти по трём возможным условиям:

1. если текущая ячейка имеем соседние ячейки не посещёнными, поступает следующим образом:
    - проталкиваем текущую ячейку в stack, далее выбираем случайным образом одну из соседних ячеек, убираем стенку между данной ячейкой и выбранной соседней, после чего выбранную соседнюю ячейку делаем текущей и отмечаем, как уже посещённую.
2. Иначе, если stack не является пустым, производим следующие шаги:
    - выдёргиваем ячейку из stack, и делаем её текущей.
3. Иначе, при иных условиях:
    - выбираем случайным образом не посещённую ячейку, делаем её текущей и отмечаем как посещённую.

Третье условие приводится в качестве исключения, поскольку при нормальной работе данного алгоритма, а также при наличие правильных исходных данных, данное условие никогда не выполнится. Если же данное условие всё-таки выполнится, то сгенерированный алгоритм, скорее всего, будет иметь изолированную область.

В ходе создания данного приложения для формирования практических навыков были изучены необходимые теоретические материалы, содержащие возможные пути решения поставленной задачи.

Данное приложение написано на языке C++ с использованием стандартной библиотеки шаблонов STL. C++ фактически превратился в стандарт объектно-ориентированного программирования после своего появления, потому возникла потребность в его стандартизации. Данный процесс включал в себя разработку библиотек C++, которые позволяют расширить базовые возможности языка, а также содержит ряд компонентов общего назначения. Стандартная библиотека предоставляет в распоряжение набор общих классов и содержит:

- классы ввода-вывода;
- различные алгоритмы (как например, алгоритмы сортировки данных);
- классы для представления числовых данных;
- строковые типы;
- различные структуры данных (как например, связанные списки, динамические массивы, бинарные деревья);
- классы, обеспечивающие интернационализацию программ.

Все эти перечисленные возможности доступны через вполне простой программный интерфейс. Стандартные компоненты чрезвычайно важны для многих программ, поскольку обработка данных обычно сопряжена с вводом, обработкой и выводом больших объёмов данных, которые часто представляются в текстовом формате.

Библиотека STL содержит пять основных видов компонентов:

- алгоритм (определяет вычислительную процедуру) – algorithm;
- контейнер (управляет набором объектов в памяти) – container;
- итератор (обеспечивает для алгоритма средство доступа к содержимому контейнера) – iterator;
- функциональный объект (инкапсулирует функцию в объекте для использования другими компонентами) – function object;
- адаптер (адаптирует компонент для обеспечения различного интерфейса) – adaptor.

STL имеет ряд преимуществ, за счёт которых является оптимальным использование данной библиотеки. Из основных преимуществ, это: универсальность, то есть независимость от типов данных; увеличение скорости написания программ; оптимальное использование памяти; гарантированное отсутствие ошибок.

Создавая приложение оптимальным решением было использование класса stack из стандартной библиотеки STL ввиду того, что элементы stack можно добавлять, проверять или удалять только из верхней части stack, которая является последним элементом базового контейнера, тем самым этот класс предоставляет ограничение функциональности, ограничивая доступ к элементу, который был добавлен последним.

Список использованных источников.

1. Род Хаггарти, Дискретная математика для программистов // Правообладатель: Техносфера // Переводчики А.А. Ковалёв, В.А. Головешкин, В. Ульянов, С.А. Кулешов // — 2012 г. — 401 с.
2. Томас Х. Кормен, Алгоритмы. Вводный курс // Издательство: Вильямс // Переводчик Игорь Красиков — 2014 г. — 208 с.
3. Николай Джосьютис, C++ Стандартная библиотека. Для профессионалов // Издательство: Питер // — 2004 г. — 720 с.

РАЗРАБОТКА ИНТЕРНЕТ-ПОРТАЛА «ОСНОВЫ ПРОГРАММИРОВАНИЯ»

Авторы: Макарова Анастасия Владимировна, Халиков Анис Рафаэльевич студенты II курса филиала «Протвино» государственного бюджетного образовательного учреждения высшего образования Московской области «Университет «Дубна».

Научный руководитель: к.т.н., доцент Астафьева Марина Петровна

Аннотация

В работе продемонстрированы этапы создания интернет - портала и примеры его отличия от обычного сайта. Наиболее подробно в статье рассматривается теория создания интернет - портала. Задача - показать новые формы получения образования, способы саморазвития и их преимущества.

The article demonstrate the stages of creating an Internet portal and examples its of difference from a regular site. The theory of creating an Internet portal show the most detailed in the article. The task is to show new forms of education, ways of self-development and their advantages.

Неотъемлемой частью жизни современного человека является интернет. Если сначала он задумывался для использования в военных или рабочих целях, то сейчас область его применения значительно расширилась. В наши дни интернет используют для общения, развлечений, его первоначальная функция - информационная тоже сохранилась и является главной. Ежедневно мы узнаём что-либо новое, будь то мировые или местные новости, прогноз погоды, новый термин из программирования,