Федеральное агентство связи  
Федеральное государственное бюджетное  
образовательное учреждение высшего образования  
 «Сибирский государственный университет  
телекоммуникаций и информатики»

Кафедра вычислительных систем

Курсовая работа

по дисциплине «Технологии разработки программного обеспечения»

на тему “игра «Ним» (три кучки спичек)

Выполнил:

ст. гр. ИВ-122 Домбрачев И.А

Прокопьев К.С

Проверил:

ст. преподаватель Токмашева Е. И.

Содержание

[Введение и постановка задачи 3](#__RefHeading___Toc132_1766323042)

[Техническое задание 4](#__RefHeading___Toc134_1766323042)

[Описание выполненного проекта 5](#__RefHeading___Toc136_1766323042)

[Функции 5](#__RefHeading___Toc150_1766323042)

[Тесты 9](#__RefHeading___Toc152_1766323042)

[Личный вклад в проект 10](#__RefHeading___Toc138_1766323042)

[Приложение. Текст программы 11](#__RefHeading___Toc140_1766323042)

# Введение и постановка задачи

Работая в команде и используя GitHub создать законченный программный продукт.

Создать игру “Ним”

* 1. Сформулировать техническое задание (ТЗ) — документ, содержащий набор требований к проекту.

На первой итерации ТЗ должно включать как минимум следующие пункты.

1. Функциональность проекта. Описание с точки зрения пользователя: какие задачи решает продукт, какие покрывает сценарии использования.
2. Формат входных данных.
3. Интерфейс приложения. В каком режиме работает приложение (интерактивный или нет, фоновый процесс, сетевой сервис и т. д.). Какие элементы интерфейса предусмотрены, их поведение.
4. Если приложение принимает аргументы командной строки, то их формат и описание.
5. Если предполагается использовать чтение исходных данных извне программы: конфигурационного файла, базы данных, источников в Интернет и т.д., то необходимо описание формата / протокола взаимодействия.

# Техническое задание

**Игра "Ним"**

**Функциональность проекта:**

Из трёх куч, каждая из которой содержит случайное количество спичек, двое участников (игрок и компьютер), поочерёдно забирают спички, разрешается брать не меньше одной спички за ход из одной кучи, брать спички из нескольких куч за один ход нельзя. Победителем считается тот, кто забрал последнюю спичку.

**Формат входных данных:**

Работа происходит при взаимодействии с интерфейсом продукта.

Всё управление программы происходит с помощью ввода с клавиатуры.

Интерфейс приложения:

Данная программа запускается с терминала и работает в интерактивном режиме при помощи указанных в ней клавиш управления.

**Алгоритм игры компьютера:**

Ход компьютера зависит от ним-суммы, если она равна нулю возьмет одну спичку, если же ним-сумма не нулевая то возьмет необходимое количество спичек для обнуления ним-суммы.

**Интерфейс приложения:**

Приложение интерактивное, запускается в терминале.

**Элементы управления:**

Начать игры: клавиша – “n”.

Закрыть игру: любая другая клавиша.

Выбор права первого хода: компьютер – клавиша “n”, игрок – любая другая клавиша.

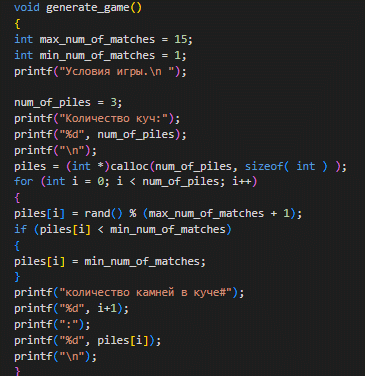
Выбрать кучу: клавиши от 1 до 3.

Выбрать число спичек: клавиша от 0 до 9.

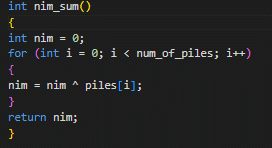
# Описание выполненного проекта

**1. Функции**

Функция создает условия игры, генерируя случайное количество спичек в каждой куче, нажатием клавиши ‘n’ можно генерировать новое условие.



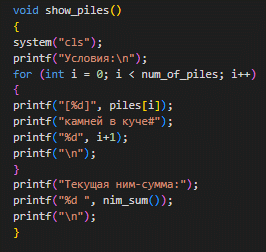
Функция возвращает ним-сумму на основе сгенерированного условия. **(ним-сумма** результат сложения размеров всех кучек в двоичной системе счисления без учёта переноса разрядов)



Функция выбора первого хода, при нажатии клавиши ‘n’ первый ход переходит компьютеру, любая другая нажатая клавиша отдает право первого хода игроку.

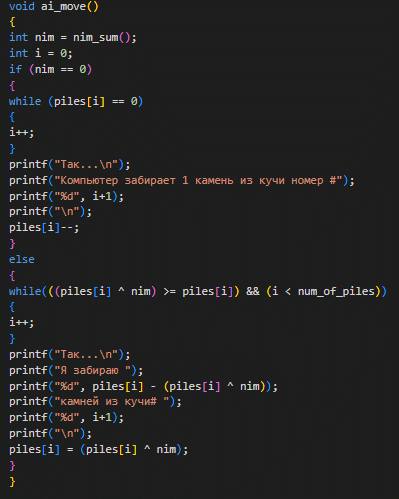


Функция выводит данные о числе спичек в кучах после каждого сделанного хода.

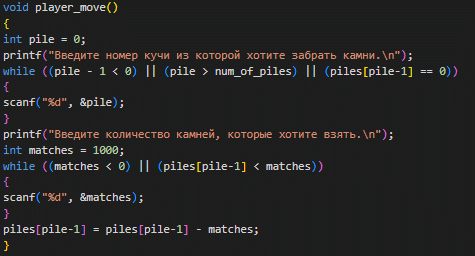


Функция работы алгоритма компьютера на основе ним-суммы.

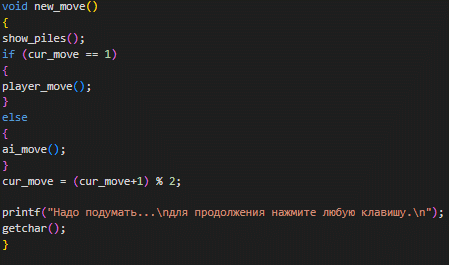
Выигрышная стратегия состоит в том, чтобы оставлять после своего хода позицию с ним-суммой, равной нулю. Она основана на том, что из любой позиции с ним-суммой, не равной нулю, можно одним ходом получить позицию с нулевой ним-суммой, а из позиции с нулевой ним-суммой любой ход ведёт в позицию с ним-суммой, отличной от нуля.



Функция ходов игрока, сначала производится выбор кучи, затем число спичек которое хочет забрать игрок.



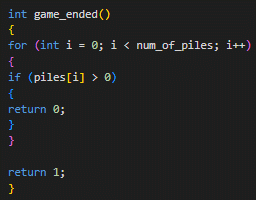
Функция передачи хода



Функция завершения игры, проверяет остались ли спички в кучах.

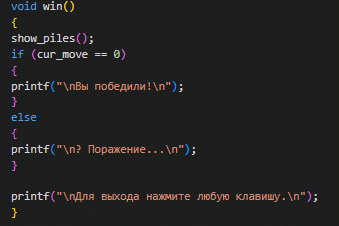
Возвращает 0 если есть хоть одна спичка, хоть в одной куче.

Возвращает 1 если ни в одной куче не осталось спичек.



# Функция определения победителя.

Если игрок возьмет последнюю спичку то программа выведет “Вы победили!”. В противном случае “Поражение…”.



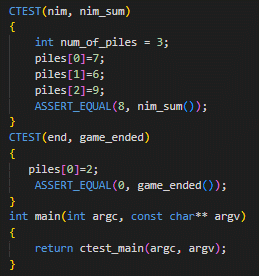
**2 Тесты.**

Тестами были покрыты следующие функции:

//game.c

nim\_sum()

game\_ended()



# Личный вклад в проект

Домбрачев И.А

* Составление ТЗ
* Разработка плана работы
* Покрытие тестами
* Функция обновления условия

Прокопьев К.С

* Написание основной части кода
* Покрытие тестами
* Реализация выбора порядка хода
* Функции выбора победителя, ходов игрока и компьютера

# Приложение. Текст программы

**Game.c**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189  190 | #include "game.h"  **int** num\_of\_piles = **3**;  **int**\* piles;  **int** cur\_move;  **void** **generate\_game**()  {  **int** max\_num\_of\_matches = **15**;  **int** min\_num\_of\_matches = **1**;  printf("Условия игры.**\n** ");  num\_of\_piles = **3**;  printf("Количество куч:");  printf("%d", num\_of\_piles);  printf("**\n**");  piles = (**int** \*)calloc(num\_of\_piles, **sizeof**( **int** ) );  **for** (**int** i = **0**; i < num\_of\_piles; i++)  {  piles[i] = rand() % (max\_num\_of\_matches + **1**);  **if** (piles[i] < min\_num\_of\_matches)  {  piles[i] = min\_num\_of\_matches;  }  printf("количество спичек в куче#");  printf("%d", i+**1**);  printf(":");  printf("%d", piles[i]);  printf("**\n**");  }  }  **int** **nim\_sum**()  {  **int** nim = **0**;  **for** (**int** i = **0**; i < num\_of\_piles; i++)  {  nim = nim ^ piles[i];  }  **return** nim;  }  **void** **choose\_first\_move**()  {  system("cls");  printf("Выберите кто будет ходить первым**\n**");  printf("Нажмите 'n' и первым пойдем компьютер, любая другая клавиша и первым ходит игрок**\n**");  **if** (getchar() == 'n')  {  **if** (nim\_sum() == **0**)  {  printf("Подтвердите выбор нажатием клавиши 'n'.**\n**");  **if** (getchar() == 'n')  {  printf("Начинает компьютер.**\n**");  cur\_move = **0**;  }  **else**  {  printf("Начинает игрок.**\n**");  cur\_move = **1**;  }  }  **else**  {  printf("Начинает компьютер.**\n**");  cur\_move = **0**;  }  }  **else**  {  printf("Начинает игрок.");  cur\_move = **1**;  }  printf("Начнем играть!**\n**Для продолжения нажмите любую клавишу.**\n**");  getchar();  }  **void** **show\_piles**()  {  system("cls");  printf("Условия:**\n**");  **for** (**int** i = **0**; i < num\_of\_piles; i++)  {  printf("[%d]", piles[i]);  printf("спичек в куче#");  printf("%d", i+**1**);  printf("**\n**");  }  printf("Текущая ним-сумма:");  printf("%d ", nim\_sum());  printf("**\n**");  }  **void** **ai\_move**()  {  **int** nim = nim\_sum();  **int** i = **0**;  **if** (nim == **0**)  {  **while** (piles[i] == **0**)  {  i++;  }  printf("Так...**\n**");  printf("Компьютер забирает 1 камень из кучи номер #");  printf("%d", i+**1**);  printf("**\n**");  piles[i]--;  }  **else**  {  **while**(((piles[i] ^ nim) >= piles[i]) && (i < num\_of\_piles))  {  i++;  }  printf("Так...**\n**");  printf("Я забираю ");  printf("%d", piles[i] - (piles[i] ^ nim));  printf("спичек из кучи# ");  printf("%d", i+**1**);  printf("**\n**");  piles[i] = (piles[i] ^ nim);  }  }  **void** **player\_move**()  {  **int** pile = **0**;  printf("Введите номер кучи из которой хотите забрать спички.**\n**");  **while** ((pile - **1** < **0**) || (pile > num\_of\_piles) || (piles[pile-**1**] == **0**))  {  scanf("%d", &pile);  }  printf("Введите количество спичек, которые хотите взять.**\n**");  **int** matches = **1000**;  **while** ((matches < **0**) || (piles[pile-**1**] < matches))  {  scanf("%d", &matches);  }  piles[pile-**1**] = piles[pile-**1**] - matches;  }  **void** **new\_move**()  {  show\_piles();  **if** (cur\_move == **1**)  {  player\_move();  }  **else**  {  ai\_move();  }  cur\_move = (cur\_move+**1**) % **2**;  printf("Надо подумать...**\n**для продолжения нажмите любую клавишу.**\n**");  getchar();  }  **int** **game\_ended**()  {  **for** (**int** i = **0**; i < num\_of\_piles; i++)  {  **if** (piles[i] > **0**)  {  **return** **0**;  }  }  **return** **1**;  }  **void** **win**()  {  show\_piles();  **if** (cur\_move == **0**)  {  printf("**\n**Вы победили!**\n**");  }  **else**  {  printf("**\n**? Поражение...**\n**");  }  printf("**\n**Для выхода нажмите любую клавишу.**\n**");  } |

}

**Main.c**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | #include "game.h"  **extern** piles;  **int** **main**()  {  printf("Я игра НИМ.**\n**Для начала игры нажмите'n'.**\n**Для закрытия нажмите любую другую клавишу.**\n**");  **char** choice = getchar();  **if** (choice != 'n')  {  **return** **1**;  }  **while** (choice == 'n')  {  free(piles);  system("cls");  generate\_game();  printf("Если вы хотите изменить условия нажмите 'n'./n");  choice = getchar();  }  choose\_first\_move();  **while**(!game\_ended())  {  new\_move();  }  win();  free(piles);  getchar();  **return** **0**;  } |

**Game.h**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | #ifndef GAME\_H  #define GAME\_H  #include <stdlib.h>  #include <stdio.h>  **void** **generate\_game**();  **int** **nim\_sum**();  **void** **choose\_first\_move**();  **void** **show\_piles**();  **void** **ai\_move**();  **void** **player\_move**();  **void** **new\_move**();  **int** **game\_ended**();  **void** **win**();  **int** **main**();  #endif |