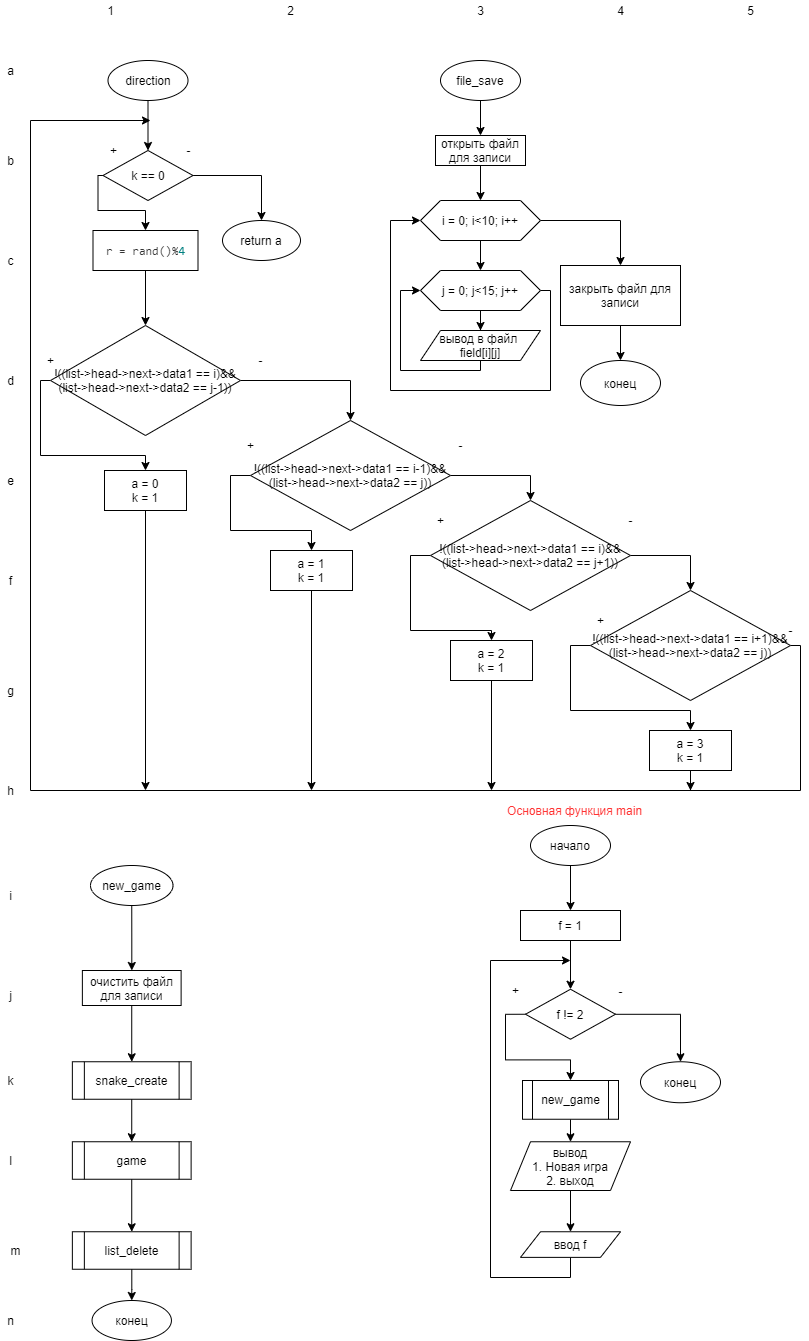
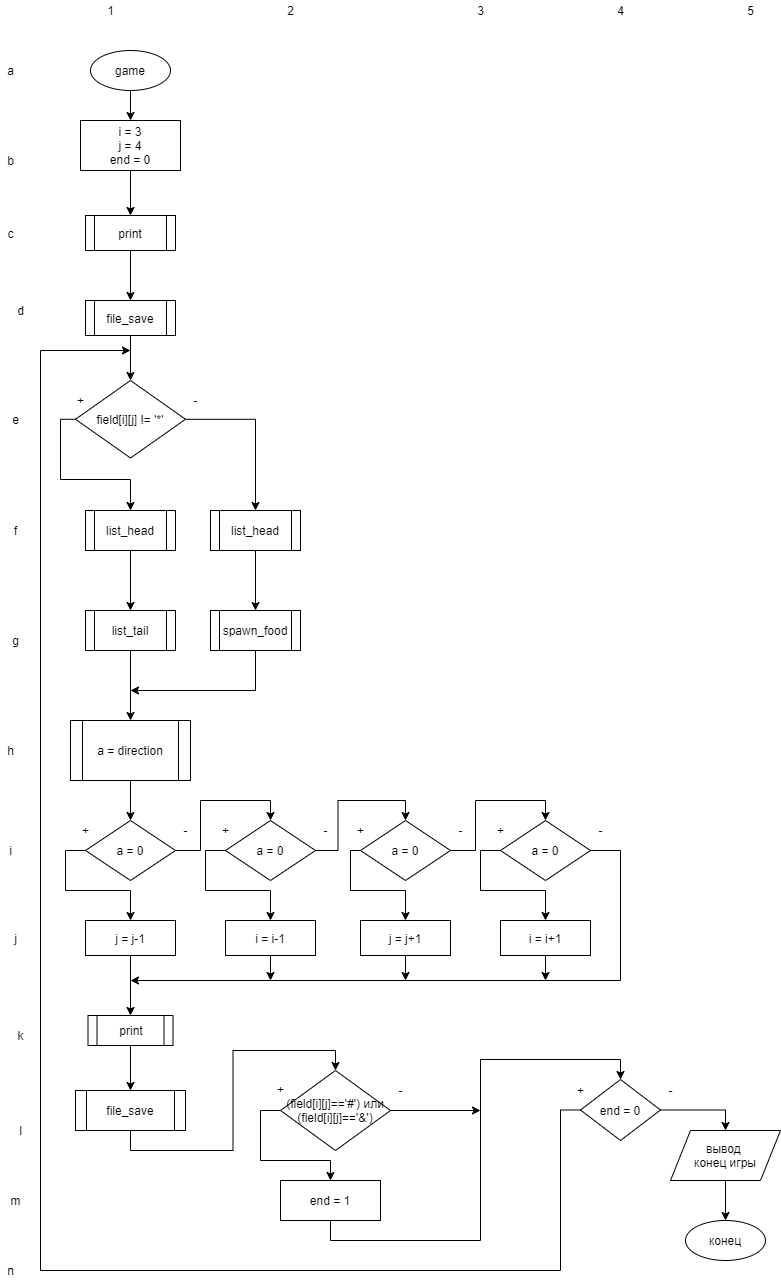
Самородов Роман Б20-513

6.15 Реализовать игру в «Змейку» («Snake»), в которой программа самостоятельно будет играть за игрока (без интерактивности), используя генератор случайных чисел для определения его действий. Игровое поле должны отображаться в терминале постоянно обновляться. Препятствия и тело змейки должны описываться в виде списков. Файл должен использоваться для записи хода игры, возможностью его воспроизведения

**1. Блок-схема алгоритма работы основной программы и функций**

****

****

**2. Исходный код программы.**

**Lab6.c**

int main()

{

int f = 1;

srand(time(NULL));

while (f != 2)

{

new\_game();

printf("\n1. Новая игра\n2. Выход\n");

scanf("%d", &f);

}

return 0;

}

**Lab6\_1.h**

typedef struct item {

int data1;

int data2;

struct item \*next;

} item;

typedef struct list {

item \*head;

item \*tail;

} list;

list \*list\_new() {

return (list \*) calloc(1, sizeof(list));

}

void list\_delete(list \*list)//удаление списка

{

item \*ptr = list->head, \*ptr\_prev;

while (ptr)

{

ptr\_prev = ptr;

ptr = ptr->next;

free(ptr\_prev);

}

free(list);

}

int list\_put(list \*list, int data1, int data2)//добавление элемента в конец списка

{

item \*new = (item \*) calloc(1, sizeof(item));

if (!new)

{

return -1;

}

new->data1 = data1;

new->data2 = data2;

new->next = NULL;

if (!list->head)

{

list->head = new;

list->tail = new;

} else {

list->tail->next = new;

list->tail = new;

}

return 0;

}

int list\_put\_head(list \*list,int data1,int data2)//поместить элемент в начало списка

{

item \*new = (item \*) calloc(1, sizeof(item));

if (!new)

{

return -1;

}

new->data1 = data1;

new->data2 = data2;

new->next = list->head;

list->head = new;

return 0;

}

int list\_head(list \*list, char field[10][15], int data1, int data2)//перемещение головы змейки

{

list\_put\_head(list, data1, data2);

field [list->head->data1][list->head->data2] = '@';

field [list->head->next->data1][list->head->next->data2] = '&';

return 0;

}

int list\_tail(list \*list, char field[10][15])//перемещение хвоста змейки

{

int i, j;

item \*tmp = list->head;

while (tmp->next != list->tail)

{

tmp = tmp->next;

}

i = tmp->next->data1;

j = tmp->next->data2;

field[i][j] = ' ';//освобождение ячейки поля

list->tail = tmp;

tmp = tmp->next;

free(tmp);

list->tail->next = NULL;

return 0;

}

int print(char field[10][15])//вывод поля игры

{

printf("\n");

for(int i = 0; i < 10; i++)

{

for(int j = 0; j < 15; j++)

printf("%c", field[i][j]);

printf("\n");

}

return 0;

}

int spawn\_food(char field[10][15])//генерация еды

{

int i, j, k = 0;

while (k == 0)

{

i = rand()%10;

j = rand()%15;

if (field[i][j] == ' ')

{

field[i][j] = '\*';

k = 1;

}

}

return 0;

}

int snake\_create(list\* list)//задание начального положения змейки

{

list\_put(list, 3, 3);

list\_put(list, 3, 2);

list\_put(list, 3, 1);

return 0;

}

int direction(list \*list, char field[10][15], int i, int j)//выбор направления

{

int r, a, k = 0;

while (k == 0)

{

r = rand()%4;

if (r == 0)//влево

{

if(!((list->head->next->data1 == i)&&(list->head->next->data2 == j-1)))

{

a = 0;

k = 1;

}

}

else if (r == 1)//вверх

{

if(!((list->head->next->data1 == i-1)&&(list->head->next->data2 == j)))

{

a = 1;

k = 1;

}

}

else if (r == 2)//вправо

{

if(!((list->head->next->data1 == i)&&(list->head->next->data2 == j+1)))

{

a = 2;

k = 1;

}

}

else if (r == 3)//вниз

{

if(!((list->head->next->data1 == i+1)&&(list->head->next->data2 == j)))

{

a = 3;

k = 1;

}

}

}

return a;

}

int file\_save(char field[10][15])

{

FILE \*in;

in = fopen("filein.txt", "a");

if(in == NULL)

{

printf("\nФайл не найден");

return 0;

}

for (int i=0; i<10; i++)

{

for (int j=0; j<15; j++)

fprintf(in, "%c", field[i][j]);

fprintf(in, "\n");

}

fclose(in);

return 0;

}

int file\_clean()

{

FILE \*in;

in = fopen("filein.txt", "w");

fclose(in);

return 0;

}

**Lab6\_2.h**

int game(list \*list, char field[10][15])//игровой процесс

{

int next, a, i = 3, j = 4, end = 0;//змея сначала ползет вправо к еде

print(field);

file\_save(field);

scanf("%c", &next);//нажатие любой клавиши для следущего хода

do {

if (field[i][j] != '\*')

{

list\_head(list, field, i, j);

list\_tail(list, field);

}

else {

list\_head(list, field, i, j);

spawn\_food(field);

}

a = direction(list, field, i, j); //выбор направления

if(a == 0)

{

j = j-1;//влево

}

else if(a == 1)

{

i = i-1;//вверх

}

else if(a == 2)

{

j = j+1;//вправо

}

else if(a == 3)

{

i = i+1;//вниз

}

scanf("%c", &next);//нажатие любой клавиши для следущего хода

print(field);

file\_save(field);

if ((field[i][j]=='#')||(field[i][j]=='&')) end = 1;

} while (end != 1);

printf("\nКонец игры\n");

return 0;

}//конец игры

int file\_clean()

{

FILE \*in;

in = fopen("filein.txt", "w");

fclose(in);

return 0;

}

int new\_game()//новая игра

{

list\* list = list\_new();

char field[10][15] ={{"###############"},

{"# ## #"},

{"# #"},

{"#&&@\* #"},

{"# ### #"},

{"# # #"},

{"# # #"},

{"# ## #"},

{"# ### #"},

{"###############"}};

file\_clean();

snake\_create(list);

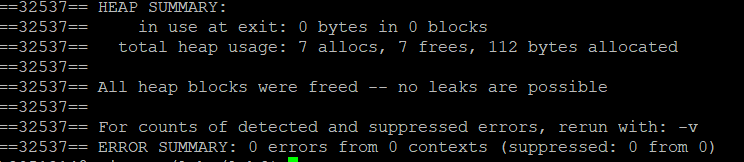
game(list, field);

list\_delete(list);

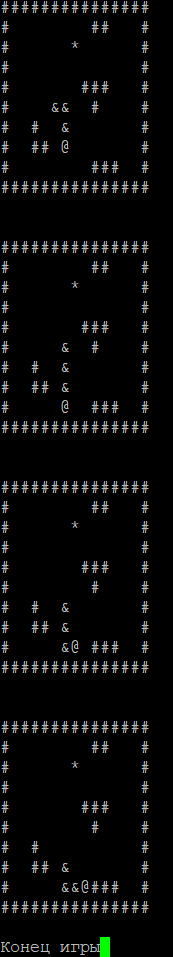
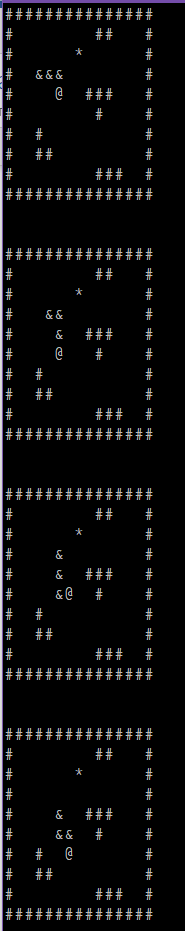
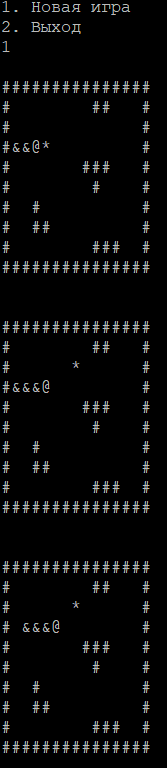
return 0;

}

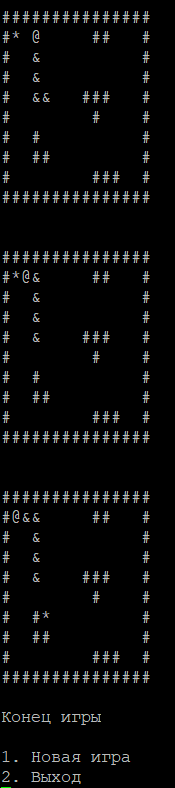
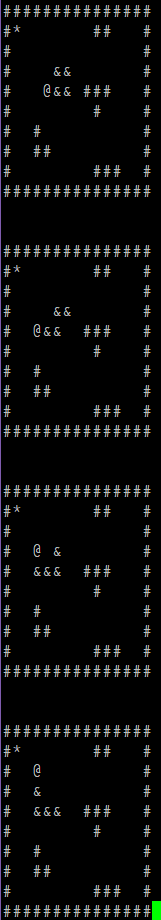
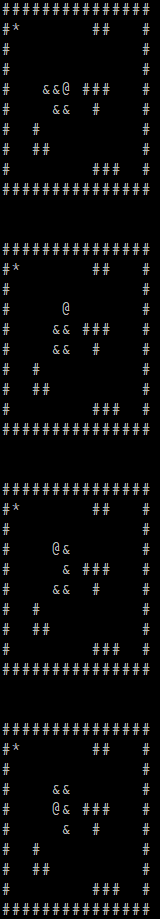
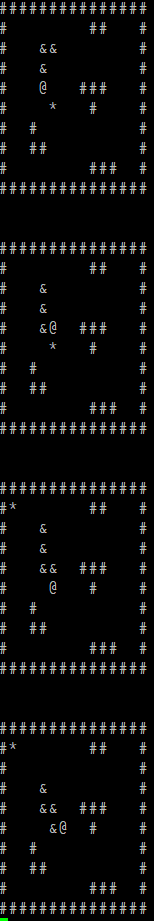
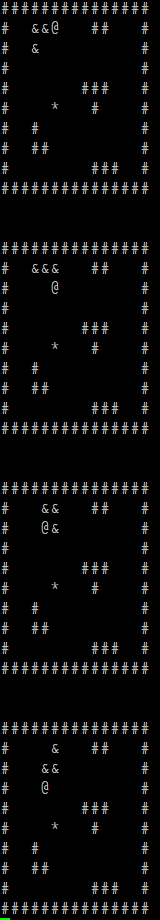
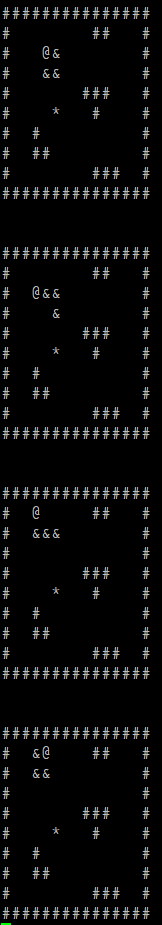
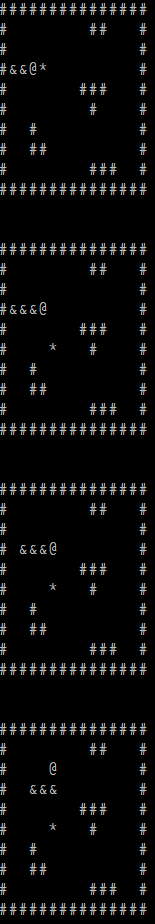
**3. Valgrind**



**4.Пример работы 1)**



**2)**



**5) Пример записи в файл**

