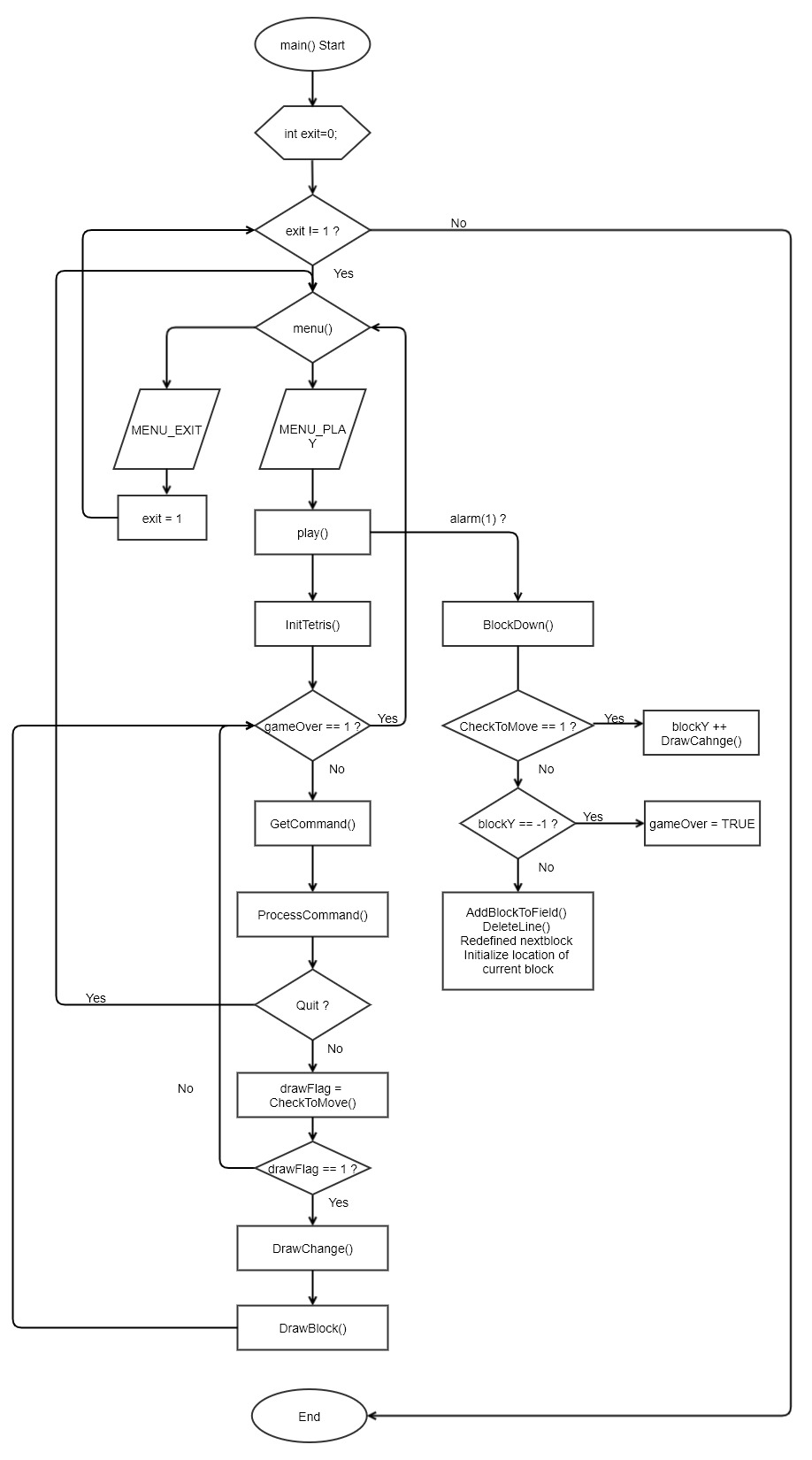
6주차 예비보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20191619 이름: 이동석

**1. Flow chart**

****

**2. 함수**

**2-1. 1주차 구현함수**

-void InitTetris() : 테트리스에 필요한 전역 변수들을 초기화 한다.

-void DrawOutline() : 테트리스의 인터페이스를 그린다.

-int GetCommand() : 4개의 방향키와 ‘q’ ‘Q’ 를 입력받는다.

-int ProcessCommand() : Getcommand()로 받은 command 동작을 실행한다.

-void BlockDown() : 블록이1초마다 아래로 한칸 씩 자동으로 내려가게 만든다. AdBlockToField(), DeleteLine(), DrawNextBlock(), DrawField() 을 호출하며, 다음블록과 현재블록을 재할당 한다.

-int CheckToMove() : 블록이 화면에 벗어나지 않고 이동할 수 있는지 확인하는 함수이다.

-void DrawChange(): 방향키로 변경된 블록을 필드에 반영한다.

-void DrawField() : 필드를 갱신한다.

-void AdBlockToField() : 필드에 블록을 쌓는다.

-int DeleteLine() : 채워진 라인이 있다면 줄을 삭제하고, 삭제한 줄 만큼 필드를 내린다.

-void gotoyx() : 현재 nucrses 라이브러리의 move로 대체 되고 있다.

-void DrawNextBlock() : 테트리스 인터페이스 상 오른쪽 박스에 나오는 블록을 그린다.

-void PrintScore() : 테트리스 인터페이스 상 오른쪽 하단에 Score를 출력한다.

-void DrawBox() : 원하는 크기의 box를 화면에 출력한다.

-void DrawBlock() : 원하는 좌표에 해당 블록을 출력한다.

-void play() : 테트리스 게임을 시작한다.

-char menu() : 게임 시작 전 menu화면을 출력한다.

**2-2. 미구현 함수**

- void DrawShadow() : 블록이 떨어질 위치를 화면에 표시한다.

-void createRankList() : 현재 순위를 rank 파일로부터 읽어와 구성한다.

-void rank() : 순위정보를 화면에 출력한다.

-void writeRankFile() : rank 파일을 생성한다.

-void newRank() : 순위에 새로운 기록을 추가한다.

-void recommend() : 추천하는 블록 배치를 찾는다.

-recommendPlay() : 블록을 위의 함수를 참고하여 블록을 배치해 게임을 진행한다.

**3. Pseudo code**

/\* timed\_out, gameOver, score are global variables \*/

int CheckToMove(char f[HEIGHT][WIDTH],int currentBlock,int blockRotate, int blockY, int blockX){

/\* return TRUE if block can move \*/

int w,h

for h=0 to 3

for w=0 to 3

if block[currentBlock][blockRotate][h][w]==1

if the block is already there return FALSE

else if out of height and width return FALSE

endif

Endfor

Endfor

return TRUE

}

void DrawChange(char f[HEIGHT][WIDTH],int command,int currentBlock,int blockRotate, int blockY, int blockX){

/\* update the field by command \*/

int w,h, preRotate

CASE OF command

KEY\_UP :

KEY\_DOWN :

KEY\_RIGHT :

KEY\_LEFT : delete the previous block

OTHER : break;

ENDCASE

DrawBlock(blockY, blockX, currentBlock, blockRotate,' ')

}

void BlockDown(int sig){

/\* blocs fall every second \*/

timed\_out=0 // To make it fall automatically

if the block can move down

blockY++

DrawChange(field, KEY\_DOWN, nextBlock[0], blockRotate, blockY, blockX)

else if the block touch the top

gameOver=TRUE

return

else

AddBlockToField(field, nextBlock[0], blockRotate, blockY, blockX)

score+=DeleteLine(field)

nextBlock[0]=nextBlock[1]

nextBlock[1]=rand()%7 // To make next block

Initialize the variables blockRotate, blockY, blockX

DrawNextBlock(nextBlock)

DrawField()

PrintScore(score)

endif

}

void AddBlockToField(char f[HEIGHT][WIDTH],int currentBlock,int blockRotate, int blockY, int blockX){

/\* add block in field \*/

int w,h

for h=0 to 3

for w=0 to 3

if block[currentBlock][blockRotate][h][w]==1

f[blockY+h][blockX+w]=1

endif

Endfor

Endfor

}

int DeleteLine(char f[HEIGHT][WIDTH]){

/\* Delete the line and raise the score \*/

int w,h,temp,flag,line=0

for h= HEIGHT-1 to 0

flag=1

for w=0 to WIDTH-1

if the line is not full

flag=0

break

Endfor

if flag==1

line++

for temp=h-1 to 0

for w=0 to WIDTH-1

f[temp+1][w]=f[temp][w]

Endfor

Endfor

for w=0 to WIDTH-1

f[0][w]=0

Endfor

h++

endif

Endfor

return line\*line\*100

}