CSE3013 (컴퓨터공학 설계 및 실험 I)
PRJ-1 테트리스 프로젝트 1주차 예비 보고서

서강대학교 컴퓨터공학과 박수현 (20181634)

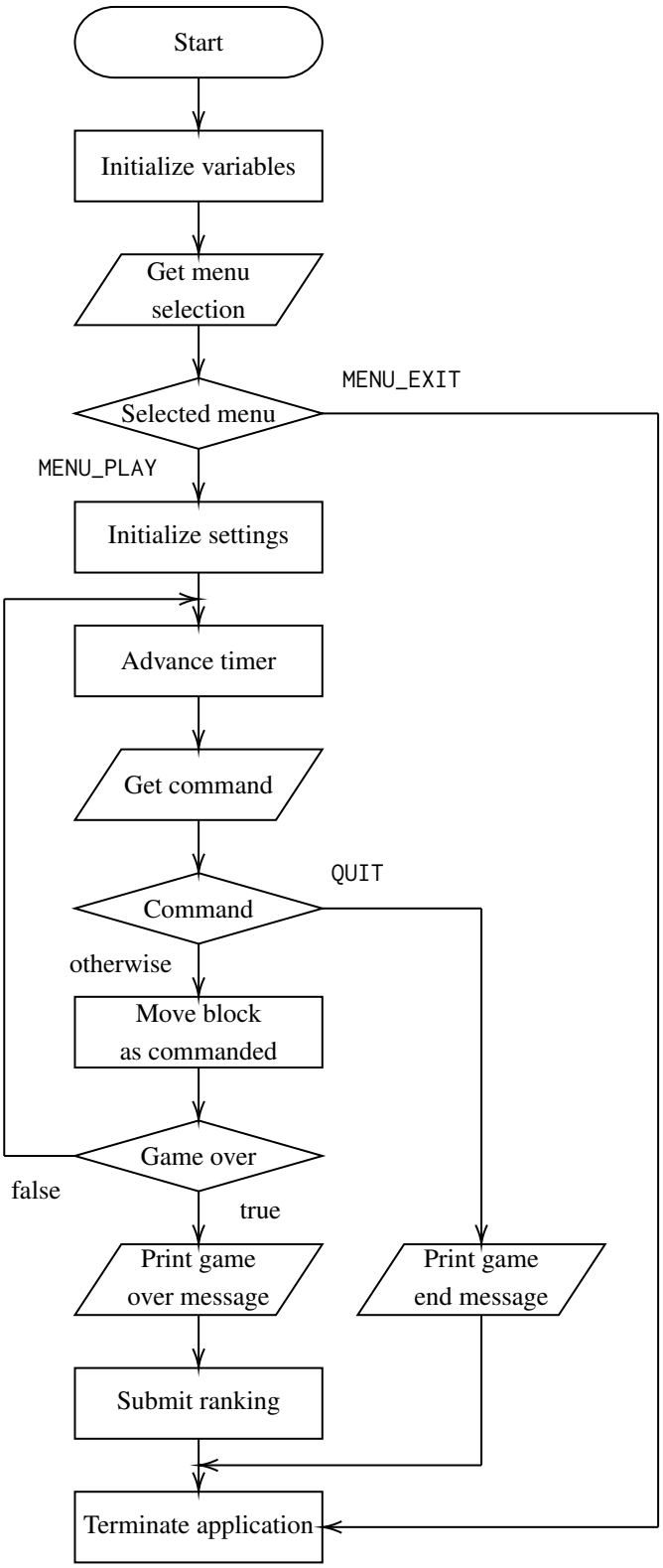
서강대학교 컴퓨터공학과

1 목적

테트리스의 구조와 테트리스를 구성하는 각 함수의 기능을 이해한다.

2 문제

2.1 테트리스 순서도



2.2 게임을 구성하는 함수들의 기능

- `void InitTetris()`: 변수들을 초기화한 후 초기 화면을 그리는 메서드를 실행한다.
- `void DrawOutline()`: 테트리스 필드, NEXT 슬롯, 점수 인디케이터 등의 테두리를 그린다.
- `int GetCommand()`: 사용자의 입력을 받아 명령으로 해석한다. 가능한 입력은 방향 키 [↑], [↓], [→], [←]와 [Space], [Q]이다.
- `int ProcessCommand(int command)`: 해석된 명령을 실행한다. 명령이 `QUIT`이었을 경우 0, 아닐 경우 1을 반환한다.
- `void BlockDown(int sig)`: 블럭을 아래로 내린다. 더 이상 내릴 수 없을 경우 현재 블럭을 필드에 고정시키고 다음 블럭을 사용한다.
- `int CheckToMove(char f[HEIGHT][WIDTH], int currentBlock, int blockRotate, int blockY, int blockX)`: 명령에 따라 블럭을 이동할 수 있는지 판단한다.
- `void DrawChange(char f[HEIGHT][WIDTH], int command, int currentBlock, int blockRotate, int blockY, int blockX)`: 명령에 의해 바뀐 부분만 필드에 업데이트한다.
- `void DrawField()`: 테트리스 필드를 그린다.
- `void AddBlockToField(char f[HEIGHT][WIDTH], int currentBlock, int blockRotate, int blockY, int blockX)`: 필드의 주어진 좌표에 현재 블럭을 고정시킨다.
- `int DeleteLine(char f[HEIGHT][WIDTH])`: 꽉 찬 줄이 있는지 체크하고, 있을 경우 줄을 지우고 스코어를 증가시킨다.
- `void gotoyx(int y, int x)`: 커서를 해당 좌표로 이동시킨다.
- `void DrawNextBlock(int *nextBlock)`: NEXT 슬롯에 다음에 나올 블럭 정보를 표시한다.
- `void PrintScore(int score)`: 점수를 표시한다.
- `void DrawBox(int y, int x, int height, int width)`: 해당 좌표에 특정 크기의 직사각형을 그린다.
- `void DrawBlock(int y, int x, int blockID, int blockRotate, char tile)`: 해당 좌표에 특정 모양의 블럭을 그린다.
- `void DrawShadow(int y, int x, int blockID, int blockRotate)`: 블럭이 떨어질 위치를 미리 보여 주는 고스트를 그린다.
- `void play()`: 게임을 시작한다.
- `char menu()`: 메뉴를 그린다.
- `void createRankList()`: 랭킹 정보를 구성한다.
- `void rank()`: 랭킹 기록들을 그린다.
- `void writeRankFile()`: 랭킹이 저장되는 데이터베이스를 생성한다.
- `void newRank(int score)`: 새 랭킹 정보를 추가한다.
- `int recommend(RecNode *root)`: 추천하는 블럭 배치를 계산한다.
- `void recommendedPlay()`: 추천 기능에 따라 블럭을 배치해나가면서 진행되는 게임을 시작한다.

2.3 함수 구현

첫 주에 구현하게 될 함수들에는 다음과 같은 함수들이 있다.

CHECKTOMOVE : 명령에 따라 블럭을 이동할 수 있는지 판단한다.

CHECKTOMOVE($f, currentBlock, blockRotate, blockY, blockX$)

```
1  for i = 0 to 3
2      for j = 0 to 3
3          if block[currentBlock][blockRotate][i][j] == 1
4              x = blockX + j
5              y = blockY + i
6              if (0 ≤ x < WIDTH and 0 ≤ y < HEIGHT) ≠ TRUE
7                  return FALSE
8              if f[y][x] == 1
9                  return FALSE
10 return TRUE
```

DRAWCHANGE : 명령에 의해 바뀐 부분만 필드에 업데이트한다.

DRAWCHANGE($f, command, currentBlock, blockRotate, blockY, blockX$)

```
1  // Erase current block
2  DRAWFIELD()
3  // Draw new block
4  DRAWBLOCK(blockY, blockX, currentBlock, blockRotate, ' ')
```

BLOCKDOWN : 블럭을 아래로 내린다. 더 이상 내릴 수 없을 경우 현재 블럭을 필드에 고정시키고 다음 블럭을 사용한다.

```
BLOCKDOWN(sig)
 1  if CHECKTOMOVE(f, currentBlock, blockRotate, blockY + 1, blockX)
 2      blockY = blockY + 1
 3      DRAWCHANGE(f, command, currentBlock, blockRotate, blockY, blockX)
 4  else
 5      if blockY == 1
 6          gameOver = TRUE
 7      else
 8          ADDBLOCKTOFIELD(f, currentBlock, blockRotate, blockY, blockX)
 9          score = score + DELETELINE(f)
10          // Generate new block
11          nextBlock[0] = nextBlock[1]
12          nextBlock[1] = (Random integer in 0 .. 6)
13          DRAWNEXTBLOCK(nextBlock)
14          blockY = −1, blockX = (Center of field)
15          DRAWFIELD()
16          PRINTSCORE(score)
```

ADDBLOCKTOFIELD : 필드의 주어진 좌표에 현재 블럭을 고정시킨다.

```
ADDBLOCKTOFIELD(f, currentBlock, blockRotate, blockY, blockX)
1  for i = 0 to 3
2      for j = 0 to 3
3          if block[currentBlock][blockRotate][i][j] == 1
4              f[blockY + i][blockX + j] = 1
5  return TRUE
```

DELETELINE : 꽉 찬 줄이 있는지 체크하고, 있을 경우 줄을 지우고 스코어를 증가시킨다.

```
DELETELINE(f)
 1  erased = 0
 2  for i = 0 to HEIGHT − 1
 3      flag = TRUE
 4      for j = 0 to WIDTH − 1
 5          if f[i][j] == 0
 6              flag = FALSE
 7      if flag == TRUE
 8          erased = erased + 1
 9          for y = i downto 1
10              for x = 0 to WIDTH − 1
11                  f[y][x] = f[y − 1][x]
12          for x = 0 to WIDTH − 1
13              f[0][x] = 0
14          i = i − 1
15  return 100 × erased^2
```