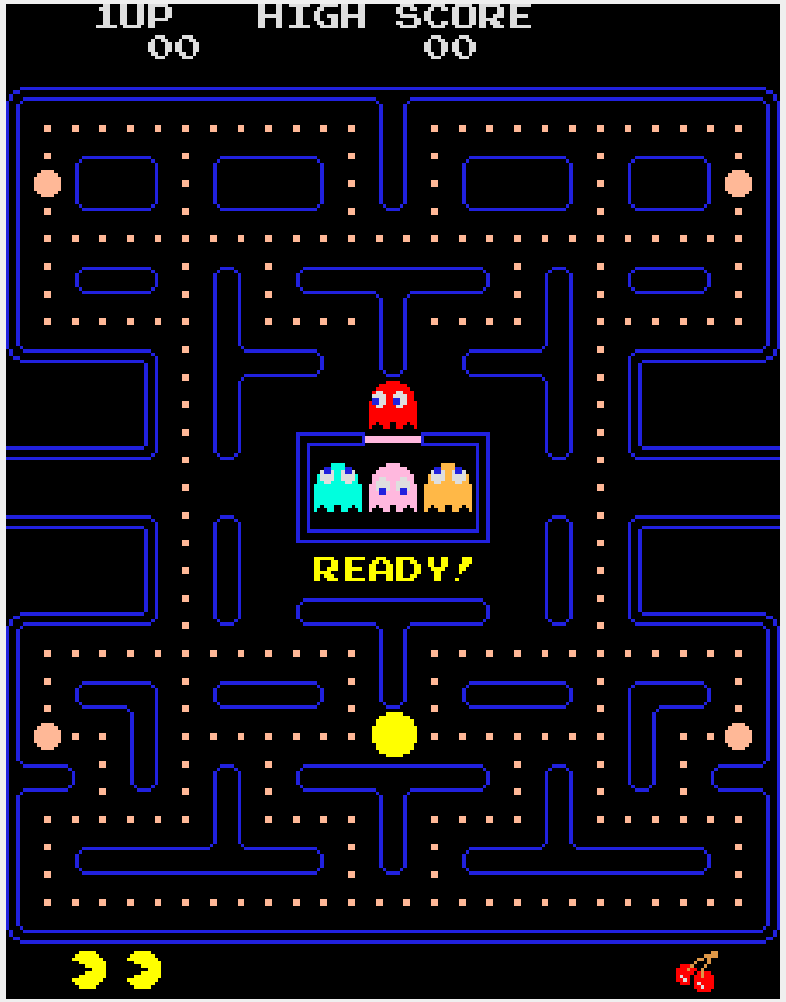
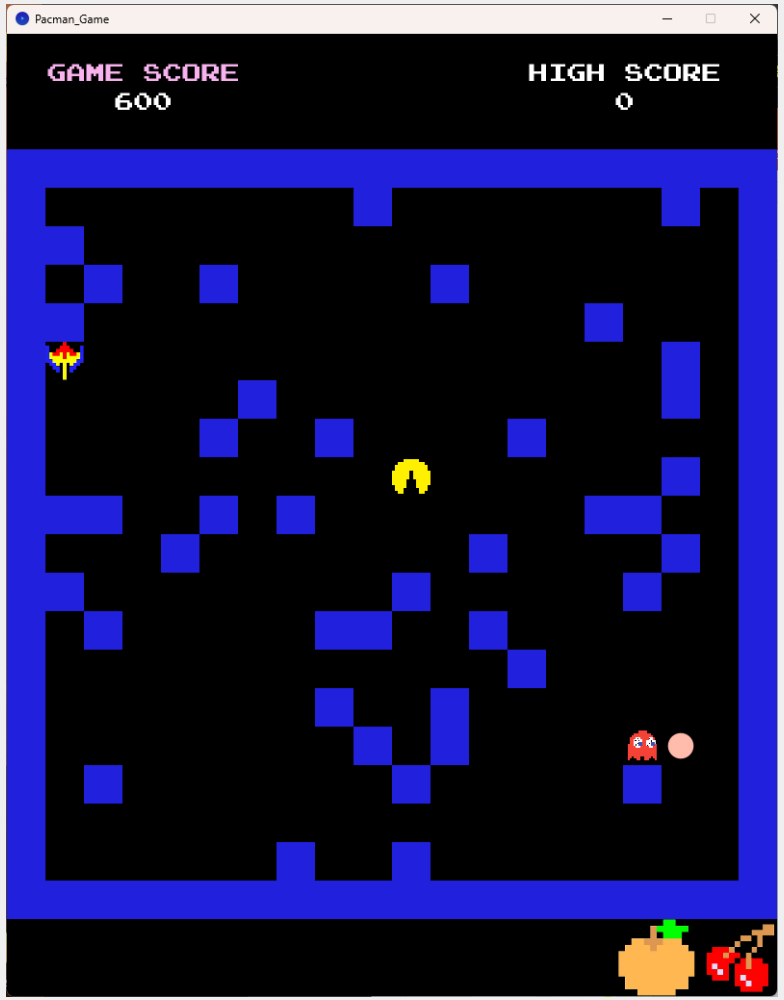
INTERACTIVE MEDIA DEV.
PACMAN GAME

CHO WOO HYUN
major; software

DIRECTION



ORIGINAL



COPY PRACTICE

기존 PACMAN처럼 스테이지 형식으로 진행되는 것이 아닌, 무한으로 랜덤위치에 생성되는 코인을 계속 먹어 높은 점수를 기록하는 게임

SETUP / DRAW

```
void setup() {
  size(800, 1000);
  CreateMap();
  pacman = new Pacman(1, 4);
  ghost = new Ghost(cols - 3, rows - 5);
  coin = new Coin();
  cherry = new Item(0, 1);
  pear = new Item(0, 2);
  ryb = new Item(0, 3);
}

int status_cherry = 0;
int status_pear = 0;
int status_ryb = 0;
```

```
void CreateMap() {
  cols = width / gridSize;
  rows = height / gridSize;
  map = new int[cols][rows];
  // 맵 초기화 - 랜덤한 벽 생성
  for (int i = 0; i < cols; i++) {
    for (int j = 0; j < rows; j++) {
      // 외곽은 항상 벽으로 설정
      if (i == 0 || i == cols - 1 || j == 3 || j == rows - 3) {
        map[i][j] = 0; // 벽
      if (j < 3 || j > rows - 3) {
        map[i][j] = 1;
      }
      } else{
        // 나머지 셀은 15%의 확률로 벽으로 설정
        if (random(1) < 0.15 && j > 3 && j < rows - 3) {
          map[i][j] = 0; // 벽
        } else {
          map[i][j] = 1; // 공간
        }
      }
    }
  }
}
```

```
void draw() {
  if (!gameOver) {
    background(0);

    //Coin을 먹었을 때
    if (pacman.x == coin.x && pacman.y == coin.y) {
      score+=100;
      coin = new Coin();
    }

    //팩맨이 움직이는 방향
    if (keystatus == 1) {
      pacman.move(0, -1);
    } else if (keystatus == 2) {
      pacman.move(0, 1);
    } else if (keystatus == 3) {
      pacman.move(-1, 0);
    } else if (keystatus == 4) {
      pacman.move(1, 0);
    }

    //Ghost와 부딪혔을 때
    if (pacman.x == ghost.x && pacman.y == ghost.y) {
      if (highscore < score) {
        highscore = score; //현재 score를 highscore로 기록
      }
      gameOver = true; //고스트와 부딪히면 gameover
    }

    //체리를 먹었을 때
    if (pacman.x == cherry.x && pacman.y == cherry.y) {
      score+=100; // 점수 증가
      cherry = new Item(1);
      //Item인자 = 오른쪽 아래에서 아이템 배열을 위한 index값
      status_cherry = 1;
    }

    //배를 먹었을 때
    if (pacman.x == pear.x && pacman.y == pear.y) {
      score+=500; // 점수 증가
      pear = new Item(2);
      status_pear = 1;
    }

    //RYB를 먹었을 때
    if (pacman.x == ryb.x && pacman.y == ryb.y) {
      score+=2000; // 점수 증가
      ryb = new Item(3);
      status_ryb = 1;
    }
```

```
    // 맵 그리기
    for (int i = 0; i < cols; i++) {
      for (int j = 0; j < rows; j++) {
        if (map[i][j] == 0) {
          fill(10, 96, 224); // 벽
        } else {
          fill(0); // 공간
          noStroke();
        }
        rect(i * gridSize, j * gridSize, gridSize, gridSize);
      }
    }
    pacman.display();
    ghost.display();
    ghost.move();
    coin.display();
    cherry.display(loadImage("image/cherry.png"), status_cherry, 0);
    pear.display(loadImage("image/pear.png"), status_pear, 1);
    ryb.display(loadImage("image/ryb.png"), status_ryb, 2);

    // 점수 표시
    font = createFont("font/pacman.ttf", 20);
    fill(248, 178, 238);
    textSize(50);
    textFont(font);
    textAlign(CENTER, CENTER);
    text("GAME SCORE ", 150, 40);
    fill(255);
    text(score, 140, 70);
    text("HIGH SCORE ", 650, 40);
    fill(255);
    text(highscore, 640, 70);
  }
  else {
    gameOver();
    restartButton();
  }
}

boolean isCellEmpty(int x, int y) {
  // 주어진 좌표가 맵 안에 있고 벽이 아니라면 true 반환
  return (x >= 0 && x < cols && y >= 0 && y < rows && map[x][y] == 1);
}
```

OBJECT

```
class Pacman {
  int x, y;

  Pacman(int x, int y) {
    this.x = x;
    this.y = y;
    photo = loadImage("image/pacman_right.png");
  }

  void move(int xdir, int ydir) {
    // 새로운 위치 계산
    int newx = x + xdir;
    int newy = y + ydir;

    // 벽에 부딪치지 않으면 이동
    if (isCellEmpty(newx, newy)) {
      x = newx;
      y = newy;
    }
  }

  void display() {
    image(photo, x * gridSize, y * gridSize, gridSize, gridSize);
  }
}
```



```
class Ghost {
  int x, y;
  int speed = 5;
  int lastMoveFrame;
  int currentDirection;

  Ghost(int x, int y) {
    this.x = x;
    this.y = y;
    lastMoveFrame = frameCount;
    currentDirection = int(random(4));
  }

  void move() {
    if (frameCount - lastMoveFrame > speed) {
      int newx = x;
      int newy = y;

      //한쪽 방향을 유지
      if (currentDirection == 0) {
        newx = x - 1; // 왼쪽
      } else if (currentDirection == 1) {
        newx = x + 1; // 오른쪽
      } else if (currentDirection == 2) {
        newy = y - 1; // 위
      } else if (currentDirection == 3) {
        newy = y + 1; // 아래
      }

      //벽을 만나면 랜덤한 방향을 선택하여 그 방향으로 계속 이동
      if (!isCellEmpty(newx, newy) || newx < 0
      || newx >= cols || newy < 0 || newy >= rows) {
        currentDirection = int(random(4));
        return;
      }
      x = newx;
      y = newy;
      lastMoveFrame = frameCount;
    }
  }
  void display() {
    gphoto = loadImage("image/ghost.png");
    image(gphoto, x * gridSize, y * gridSize, gridSize, gridSize);
  }
}
```

```
class Coin {
  int x, y;

  Coin() {
    while (true) {
    int newx = x;
    int newy = y;
    newx = int(random(1, cols - 1));
    newy = int(random(4, rows - 3));
    if (isCellEmpty(newx, newy)) {
        x = newx;
        y = newy;
        break;
      }
    }
  }
  void display() {
    fill(255, 188, 172);
    ellipse(x * gridSize + gridSize/2, y * gridSize + gridSize/2, gridSize / 1.5, gridSize / 1.5);
  }
}
```

```
class Item {
  int x, y;
  int upsize;
  int volume;

  Item(int vol) {
    volume = vol;
    while (true) {
    int newx = x;
    int newy = y;
    newx = int(random(1, cols - 1));
    newy = int(random(4, rows - 4));
    if (isCellEmpty(newx, newy)) {
        x = newx;
        y = newy;
        break;
      }
    }
  }

  void display(PImage items, int status, int volume) {
    if (status == 0) {
      image(items, x * gridSize, y * gridSize, gridSize, gridSize);
    }
    else if (status == 1) {
      image(items, 18 * gridSize + (volume * -85), 23 * gridSize, gridSize * 2, gridSize * 2);
    }
  }
}
```

SYSTEM

```
void gameOver() {
  fill(90, 173, 255);
  textSize(40);
  text("PLAYER", width / 2, (height / 2) + -50);
  fill(113, 17, 0);
  text("GAME OVER", width / 2, (height / 2) + 50);
  noLoop();
}

void restartButton() {
  fill(50, 50, 50);
  rect(width / 2 - 100, height / 2 + 150, 200, 50);
  fill(255);
  textSize(25);
  text("RESTART", width / 2, height / 2 + 175);
}

void mousePressed() {
  if (gameOver == true && mouseX > width / 2 - 100 && mouseX < width / 2 + 100 &&
      mouseY > height / 2 + 150 && mouseY < height / 2 + 200) {
    resetGame();
    gameOver = false;
  }
}
```

```
//RESTART버튼 누를시 맵을 새로 생성하고 모든 객체 초기화, 점수는 0으로 초기화
void resetGame() {
  CreateMap();
  score = 0;
  pacman = new Pacman(1, 4);
  ghost = new Ghost(cols - 3, rows - 5);
  coin = new Coin();
  status_cherry = 0;
  status_pear = 0;
  status_ryb = 0;
  keystatus = 0;
  cherry = new Item(1);
  pear = new Item(2);
  ryb = new Item(3);
  loop();
}
```

```
int keystatus = 0;
void keyPressed() {
  if (keyCode == UP) {
    photo = loadImage("image/pacman_top.png");
    keystatus = 1;
  } else if (keyCode == DOWN) {
    photo = loadImage("image/pacman_down.png");
    keystatus = 2;
  } else if (keyCode == LEFT) {
    photo = loadImage("image/pacman_left.png");
    keystatus = 3;
  } else if (keyCode == RIGHT) {
    photo = loadImage("image/pacman_right.png");
    keystatus = 4;
  } else if (key == 'R' || key == 'r') {
    resetGame(); //R키 누르면 리셋
  }
}
```