Create – Applications From Ideas
Written Response Submission Template

Please see Assessment Overview and Performance Task Directions for Student for the task directions and recommended word counts.

Program Purpose and Development

2a)

My program, 'FrogFly,' is a game created with JavaScript in the Code.org App Lab. The purpose of this program is to be a fun clicker-game for users to play. The goal of this game is to get as many points by clicking the animal. The video shows key features of my program, such as buttons to navigate throughout different screens, score increase, lives reduction. In the beginning, 3 lives are given but the player loses a life when they click elsewhere other than the animal. The player can buy items in the store using the coins they have earned. In the video, I get a butterfly from the store. The program restarts when the player loses all of their lives.

2b)

This program 'FrogFly' game is an extension of a clicker-game that I built in class and developed solely by me. I started by creating multiple screens and designing the backgrounds, font styles, buttons. Then I added onEvent handlers to the buttons to navigate through the app. After, I added transparent background images to make the user interface look decent. One major difficulty I encountered while developing this program is how to automatically make the animal move faster gradually. I figured it out by adding a timedLoop() function - executes the function every time a certain number of milliseconds have elapsed - and placing the move() in it. Opportunities I recognized during the development are to add a store where the player can buy items, and a pause button where the player can resume the game, access the store or quit the game.

2c)

```
//Main Algorithm
function play() {
  update();
  timer(time);
  //Increase the score by clicking the animal
  onEvent(object, "click", function() {
    move();
    score = score + multiplier;
    update();
    stopTimedLoop();
    //The animal moves faster with every click
    time = time - 10;
    timer(time);
  });
  //Lose life by clicking the background
  onEvent("backGround", "click", function() {
    lives = lives - 1;
    update();
  });
}

//Changes the animal position
function move(){
  if (object == "frog") {
    setPosition(object, randomNumber(0,220), randomNumber(245, 310));
  } else if (object == "butterfly"){
    setPosition(object, randomNumber(0,220), randomNumber(75, 310));
  }
}

//Moves the animal automatically
function timer(num) {
  timedLoop(num, move);
}
```

```
//Refreshes the User Interface
function update(){
  setProperty("score_lbl", "text",("💰"+score));
  setText("multipliere","x" + multiplier);
  setProperty("number_Points", "text",("💰"+score));

  if (lives == 3) {
    showElement("life1");
    showElement("life2");
    showElement("life3");
  } else if (lives == 2){
    hideElement("life1");
    showElement("life2");
    showElement("life3");
  } else if (lives == 1){
    hideElement("life1");
    hideElement("life2");
    showElement("life3");
  } else if (lives == 0){
    hideElement("life1");
    hideElement("life2");
    hideElement("life3");
    setScreen("lose_screen");
  }
}
```

My main algorithm is inside the function 'play()', which is developed independently by me. I put onEvent handlers in this function to implement the sequence of code when the animal is clicked. This allows the player to play the game, clicking the animal and increase the score. This function also makes the player lose a life when they click elsewhere. This algorithm achieves the intended purpose of my overall program, which is score increase and lives reduction. This algorithm has some sub-algorithms to make the program efficient. These include 'move()', 'timer()', and 'update()'. Each of them has an individual purpose for the program. Function ‘move()’ makes the animal move across the canvas. Function ‘timer()’ automatically moves the animal. Function ‘update()’ is crucial because it displays the values on the user interface. Combing all of the sub-algorithms made my program simpler and quicker.

2d)

```
//Main Algorithm
function play() {
  update();
  timer(time);
  //Increase the score by clicking the animal
  onEvent(object, "click", function() {
    move();
    score = score + multiplier;
    update();
    stopTimedLoop();
    //The animal moves faster with every click
    time = time - 10;
    timer(time);
  });
  //Lose life by clicking the background
  onEvent("backGround", "click", function() {
    lives = lives - 1;
    update();
  });
}

//Changes the animal position
function move(){
  if (object == "frog") {
    setPosition(object, randomNumber(0,220), randomNumber(245, 310));
  } else if (object == "butterfly"){
    setPosition(object, randomNumber(0,220), randomNumber(75, 310));
  }
}

//Moves the animal automatically
function timer(num) {
  timedLoop(num, move);
}
```

My abstraction includes in the function 'play()', which is the main part of the program that I wrote by myself. This function allows the player to increase their score, lose a life when the player clicks elsewhere. Additionally, using the onEvent handlers, when the user clicks the animal this code makes it move faster to a random location using the sub-algortihms 'move()', and 'timer()'. This allowed me to reduce the complexity by placing all the crucial parts of the program inside one function.