Laporan Tugas 2
Struktur Data dan Analisa Algoritma

Mochammad Hilmi Rusydiansyah
5024211008

September 28, 2022

Source Code Linked List with OOP

```
#include <iostream>
#include <unistd.h>
// #include <windows.h>

struct TNode{
    float data;
    TNode* next;
};

class TNode_manager{
private:
    TNode* head;
    TNode* tail;
    int length;
public:
    TNode_manager(){
        head = new TNode;
        tail = new TNode;
        length = 0;
    }

    int getLength(){
        return length;
    }

    void append(float input){
        TNode* tempNode = new TNode;
        tempNode->data = input;
        tempNode->next = NULL;

        if(head->next == NULL){  // first assignment
            head->data = tempNode->data;
            head->next = tempNode;

            tail = head;  // in the first assignment, head == tail
        } else{
            tail->next = tempNode;
            tail = tempNode;
        }
        length++;
    }

    void display(){
        TNode* temp = head;
        while(temp){
            std::cout << temp << " ==> " <<  temp->data << " -- " << temp
    ->next << std::endl;
            temp = temp->next;
        }
        std::cout << std::endl;
    }
```

```
51
52      void insert(int index, float input){
53          int i = 0;
54          TNode* tempIter = head;
55          TNode* newNode = new TNode;
56          newNode->data = input;
57
58          while(tempIter){
59              if(i == index-1){
60                  newNode->next = tempIter->next;
61                  tempIter->next = newNode;
62                  break;
63              }
64              tempIter = tempIter->next;
65              i++;
66          }
67          length++;
68      }
69
70      void remove(int index){
71          int i = 0;
72          TNode* tempIter = head;
73
74          while(tempIter){
75              if(i == index-1){
76                  tempIter->next = (tempIter->next)->next;
77                  break;
78              }
79              tempIter = tempIter->next;
80              i++;
81          }
82          length--;
83      }
84
85      int search(float target){
86          int result = -1;
87          int i = 0;
88          TNode* tempIter = head;
89
90          while(tempIter){
91              if(tempIter->data == target){
92                  result = i;
93                  break;
94              }
95              tempIter = tempIter->next;
96              i++;
97          }
98          return result;
99      }
100
101     float get(int index){
102         int i = 0;
103         TNode* tempIter = head;
```

```
104
105         while(tempIter){
106             if(i == index){
107                 return tempIter->data;
108             }
109             tempIter = tempIter->next;
110             i++;
111         }
112     }
113
114     void set(int index, float newValue){
115         int i = 0;
116         TNode* tempIter = head;
117
118         while(tempIter){
119             if(i == index){
120                 tempIter->data = newValue;
121                 break;
122             }
123             tempIter = tempIter->next;
124             i++;
125         }
126     }
127
128     float max(){
129         float maxVAl;
130         TNode* tempIter = head;
131
132         maxVAl = tempIter->data;
133         while(tempIter){
134             if(maxVAl < tempIter->data){
135                 maxVAl = tempIter->data;
136             }
137             tempIter = tempIter->next;
138         }
139
140         return maxVAl;
141     }
142
143     float min(){
144         float minVAl;
145         TNode* tempIter = head;
146
147         minVAl = tempIter->data;
148         while(tempIter){
149             if(minVAl > tempIter->data){
150                 minVAl = tempIter->data;
151             }
152             tempIter = tempIter->next;
153         }
154
155         return minVAl;
156     }
```

```
157
158     /* hmm bisa gak ya ditambahi fungsi reverse sama shift*/
159 };
160
161 int main(){
162     unsigned int sec_to_microsec = 1000000;
163     // unsigned int sec_to_milisec = 1000;
164     TNode_manager myLinkedList;
165
166     std::cout << "Add 5 elements" << std::endl;
167     // Sleep(1*sec_to_milisec);
168     usleep(1*sec_to_microsec);
169     std::cout << "please wait..." << std::endl;
170     // Sleep(2*sec_to_milisec);
171     usleep(2*sec_to_microsec);
172     myLinkedList.append(5);
173     myLinkedList.append(6);
174     myLinkedList.append(7);
175     myLinkedList.append(8);
176     myLinkedList.append(9);
177     std::cout << "adding process completed" << std::endl << std::endl;
178     // Sleep(1*sec_to_milisec);
179     usleep(1*sec_to_microsec);
180
181     // displaying linked list
182     std::cout << "length : " << myLinkedList.getLength() << std::endl;
183     myLinkedList.display();
184     // Sleep(2*sec_to_milisec);
185     usleep(2*sec_to_microsec);
186
187     // insertion
188     std::cout << "inserting 11 at index no-2" << std::endl;
189     // Sleep(1*sec_to_milisec);
190     usleep(1*sec_to_microsec);
191     myLinkedList.insert(2, 11);
192     std::cout << "insertion completed" << std::endl;
193     // Sleep(1*sec_to_milisec);
194     usleep(1*sec_to_microsec);
195     std::cout << std::endl;
196
197     // displaying linked list
198     std::cout << "length : " << myLinkedList.getLength() << std::endl;
199     myLinkedList.display();
200     // Sleep(2*sec_to_milisec);
201     usleep(2*sec_to_microsec);
202
203     // insertion
204     std::cout << "inserting 31.3 at index no-4" << std::endl;
205     // Sleep(1*sec_to_milisec);
206     usleep(1*sec_to_microsec);
207     myLinkedList.insert(4, 31.3);
208     std::cout << "insertion completed" << std::endl;
209     // Sleep(1*sec_to_milisec);
```

```
    usleep(1*sec_to_microsec);
    std::cout << std::endl;

    // displaying linked list
    std::cout << "length : " << myLinkedList.getLength() << std::endl;
    myLinkedList.display();
    // Sleep(2*sec_to_milisec);
    usleep(2*sec_to_microsec);

    // remove
    std::cout << "deleting element at index no-3" << std::endl;
    // Sleep(1*sec_to_milisec);
    usleep(1*sec_to_microsec);
    myLinkedList.remove(3);
    std::cout << "delete process completed" << std::endl;
    // Sleep(1*sec_to_milisec);
    usleep(1*sec_to_microsec);
    std::cout << std::endl;

    // displaying linked list
    std::cout << "length : " << myLinkedList.getLength() << std::endl;
    myLinkedList.display();
    // Sleep(2*sec_to_milisec);
    usleep(2*sec_to_microsec);

    // searching
    std::cout << "searching 31.3 in linked list" << std::endl;
    std::cout << "please wait..." << std::endl;
    // Sleep(2*sec_to_milisec);
    usleep(2*sec_to_microsec);
    int x = myLinkedList.search(31.3);
    if(x == -1){
        std::cout << "unfortunately, the searched value was not found" <<
     std::endl;
    } else{
        std::cout << "founded at index no. " << x << std::endl;
    }
    std::cout << "\n";
    // Sleep(2*sec_to_milisec);
    usleep(2*sec_to_microsec);

    // get
    std::cout << "get element with index no. 4" << std::endl;
    // Sleep(1*sec_to_milisec);
    usleep(1*sec_to_microsec);
    int temp = myLinkedList.get(4);
    std::cout << "myLinkedList index no-4 : " << temp << std::endl;
    // Sleep(2*sec_to_milisec);
    usleep(2*sec_to_microsec);
    std::cout << "\n";

    // search and set
    std::cout << "length : " << myLinkedList.getLength() << std::endl;
```

```
262     myLinkedList.display();
263     std::cout << "search value 8 and set it to 77.7" << std::endl;
264     // Sleep(1*sec_to_milisec);
265     usleep(1*sec_to_microsec);
266     std::cout << "please wait..." << std::endl;
267     // Sleep(2*sec_to_milisec);
268     usleep(2*sec_to_microsec);
269     myLinkedList.set(myLinkedList.search(8), 77.7);
270     std::cout << "done" << std::endl;
271     // Sleep(0.5*sec_to_milisec);
272     usleep(0.5*sec_to_microsec);
273     std::cout << "\n";
274     std::cout << "length : " << myLinkedList.getLength() << std::endl;
275     myLinkedList.display();
276     // Sleep(2*sec_to_milisec);
277     usleep(2*sec_to_microsec);
278 
279     // get max and min
280     std::cout << "maximum value : " << myLinkedList.max() << std::endl;
281     // Sleep(1.5*sec_to_milisec);
282     usleep(1.5 * sec_to_microsec);
283     std::cout << "minimum value : " << myLinkedList.min() << std::endl;
284     // Sleep(1.5*sec_to_milisec);
285     usleep(1.5 * sec_to_microsec);
286     std::cout << "\n";
287 
288     return 0;
289 }
```

Source Code for Pesawat Tembak-Tembakan

```
#include<graphics.h>
#include<conio.h>
#include<dos.h>
#include<stdio.h>
#include<cmath>

#define PI 3.14159265359

class Pesawat{
private :
    float x;
    float y;
    float sc;  // scale
    int Xmax ;  // screen x max
    int Ymax;  // screen y max
    int sty;  // determine plane speed y
    int stx;  // determine plane speed x
    float body[9][2] = {{0.5,0}, {0.5,-1.25}, {0,-1.5}, {-0.5,-1.25},
    {-0.5,0}, {-0.25,2.75}, {-1,3}, {1,3}, {0.25,2.75}};  // for drawing
    body plane
    float wing[6][2] = {{2.5,0.5}, {3,1.75}, {2.5,-0.5}, {-2.5,-0.5},
    {-3,1.75}, {-2.5,0.5}};  // for drawing wing plane

    // for bullet
    float xbullet;
    float ybullet;
    float radbullet = 5;
    int stybullet = 15;

public:
    Pesawat(){
        sc = rand()%15 + 5;
        Xmax = getmaxx();
        Ymax = getmaxy() / 2.75;
        sty = rand()%20 + 4;
        stx = rand()%20 + 4;
        x = rand()%Xmax + 10;
        y = rand()%Ymax + 10;
    }

    void Posisi(float xi, float yi){
        x = xi;
        y = yi;
    }

    void Skala(float scli){
        sc = scli;
    }

    void MoveUp(){
        y = y - sty;
```

```
49          if (y < 0){
50              y = y + Ymax;
51          }
52      }
53
54      void MoveRight(){
55          x = x + stx;
56          if (x > Xmax){
57              x = x - Xmax;
58          }
59      }
60
61      void RotatePesawat(float degree){
62          int num_bpoints = 9;
63          int num_wpoints = 6;
64          // convert degree to radian
65          float rad = degree * (PI/180);
66
67          // rotating body
68          for(int i=0; i<num_bpoints; i++){
69              float temp[2];  // for storing one-point coordinate
70              // assign to temp
71              for(int j=0; j<2; j++){
72                  temp[j] = body[i][j];
73              }
74              // convert cartesian to polar (r, theta)
75              float r = sqrt( pow(temp[0],2) + pow(temp[1],2) );
76              float theta = atan2(temp[1], temp[0]);
77
78              // adding theta by degree(radian) inputted
79              theta = theta + rad;
80
81              // convert polar to cartesian again and store it to temp
82              temp[0] = r * cos(theta);  // as x
83              temp[1] = r * sin(theta);  // as y
84
85              // return temp to body_array
86              for(int j=0; j<2; j++){
87                  body[i][j] = temp[j];
88              }
89          }
90
91          // rotating wing
92          for(int i=0; i<num_wpoints; i++){
93              float temp[2];  // for storing one-point coordinate
94              // assign to temp
95              for(int j=0; j<2; j++){
96                  temp[j] = wing[i][j];
97              }
98              // convert cartesian to polar (r, theta)
99              float r = sqrt( pow(temp[0],2) + pow(temp[1],2) );
100             float theta = atan2(temp[1], temp[0]);
101
```

```
            // adding theta by degree(radian) inputted
            theta = theta + rad;

            // convert polar to cartesian again and store it to temp
            temp[0] = r * cos(theta);  // as x
            temp[1] = r * sin(theta);  // as y

            // return temp to wing_array
            for(int j=0; j<2; j++){
                wing[i][j] = temp[j];
            }
        }
    }

    void DrawPesawat(){
        // drawing body of plane
        line((body[0][0]*sc + x), (body[0][1]*sc + y), (body[1][0]*sc + x), (body[1][1]*sc + y));
        line((body[1][0]*sc + x), (body[1][1]*sc + y), (body[2][0]*sc + x), (body[2][1]*sc + y));
        line((body[2][0]*sc + x), (body[2][1]*sc + y), (body[3][0]*sc + x), (body[3][1]*sc + y));
        line((body[3][0]*sc + x), (body[3][1]*sc + y), (body[4][0]*sc + x), (body[4][1]*sc + y));
        line((body[4][0]*sc + x), (body[4][1]*sc + y), (body[5][0]*sc + x), (body[5][1]*sc + y));
        line((body[5][0]*sc + x), (body[5][1]*sc + y), (body[6][0]*sc + x), (body[6][1]*sc + y));
        line((body[6][0]*sc + x), (body[6][1]*sc + y), (body[7][0]*sc + x), (body[7][1]*sc + y));
        line((body[7][0]*sc + x), (body[7][1]*sc + y), (body[8][0]*sc + x), (body[8][1]*sc + y));
        line((body[8][0]*sc + x), (body[8][1]*sc + y), (body[0][0]*sc + x), (body[0][1]*sc + y));

        // drawing wing of plane
        line((wing[0][0]*sc + x), (wing[0][1]*sc + y), (wing[1][0]*sc + x), (wing[1][1]*sc + y));
        line((wing[1][0]*sc + x), (wing[1][1]*sc + y), (wing[2][0]*sc + x), (wing[2][1]*sc + y));
        line((wing[2][0]*sc + x), (wing[2][1]*sc + y), (wing[3][0]*sc + x), (wing[3][1]*sc + y));
        line((wing[3][0]*sc + x), (wing[3][1]*sc + y), (wing[4][0]*sc + x), (wing[4][1]*sc + y));
        line((wing[4][0]*sc + x), (wing[4][1]*sc + y), (wing[5][0]*sc + x), (wing[5][1]*sc + y));
        line((wing[5][0]*sc + x), (wing[5][1]*sc + y), (wing[0][0]*sc + x), (wing[0][1]*sc + y));
    }

    void saveBulletPos(){
        xbullet = x;
        ybullet = y;
```

```
140     }
141
142     void DrawBullet(){
143         circle(xbullet, ybullet, radbullet);
144     }
145
146     void moveBullet(){
147         ybullet = ybullet + stybullet;
148     }
149
150     // tambahan
151     int getYBullet(){
152         return ybullet;
153     }
154 };
155
156 class PesawatLakon{
157 private :
158     float x;
159     float y;
160     float sc;  // scale
161     int Xmax ;  // screen x max
162     int Ymax;  // screen y max
163     int sty;  // determine plane speed y
164     int stx;  // determine plane speed x
165     float body[13][2] = {{1,-3}, {0.25,-3}, {0.25,-4.5}, {0,-5},
    {-0.25,-4.5}, {-0.25,-3}, {-1,-3}, {-1,1.5}, {-1.5,1.75}, {-1.25,2},
    {1.25,2}, {1.5,1.75}, {1,1.5}};  // for drawing body plane
166     float wing[6][2] = {{1,-2}, {5,0}, {4,1}, {-4,1}, {-5,0}, {-1,-2}};
    // for drawing wing plane
167
168     // for bullet
169     float xbullet;
170     float ybullet;
171     float radbullet = 5;
172     int stybullet = 30;
173
174 public:
175     PesawatLakon(){
176         sc = rand()%20 + 5;
177         Xmax = getmaxx();
178         Ymax = getmaxy();
179         sty = rand()%20 + 4;
180         stx = rand()%20 + 4;
181         x = rand()%Xmax + 10;
182         y = rand()%Ymax + 10;
183     }
184
185     void Posisi(float xi, float yi){
186         x = xi;
187         y = yi;
188     }
189
```

```
190     void Skala(float scli){
191         sc = scli;
192     }
193 
194     void MoveUp(){
195         y = y - sty;
196         if (y < 0){
197             y = y + Ymax;
198         }
199     }
200 
201     void MoveRight(){
202         x = x + stx;
203         if (x > Xmax){
204             x = x - Xmax;
205         }
206     }
207 
208     void RotatePesawat(float degree){
209         int num_bpoints = 13;
210         int num_wpoints = 6;
211         // convert degree to radian
212         float rad = degree * (PI/180);
213 
214         // rotating body
215         for(int i=0; i<num_bpoints; i++){
216             float temp[2];  // for storing one-point coordinate
217             // assign to temp
218             for(int j=0; j<2; j++){
219                 temp[j] = body[i][j];
220             }
221             // convert cartesian to polar (r, theta)
222             float r = sqrt( pow(temp[0],2) + pow(temp[1],2) );
223             float theta = atan2(temp[1], temp[0]);
224 
225             // adding theta by degree(radian) inputted
226             theta = theta + rad;
227 
228             // convert polar to cartesian again and store it to temp
229             temp[0] = r * cos(theta);  // as x
230             temp[1] = r * sin(theta);  // as y
231 
232             // return temp to body_array
233             for(int j=0; j<2; j++){
234                 body[i][j] = temp[j];
235             }
236         }
237 
238         // rotating wing
239         for(int i=0; i<num_wpoints; i++){
240             float temp[2];  // for storing one-point coordinate
241             // assign to temp
242             for(int j=0; j<2; j++){
```

```
                temp[j] = wing[i][j];
            }
            // convert cartesian to polar (r, theta)
            float r = sqrt( pow(temp[0],2) + pow(temp[1],2) );
            float theta = atan2(temp[1], temp[0]);

            // adding theta by degree(radian) inputted
            theta = theta + rad;

            // convert polar to cartesian again and store it to temp
            temp[0] = r * cos(theta);  // as x
            temp[1] = r * sin(theta);  // as y

            // return temp to wing_array
            for(int j=0; j<2; j++){
                wing[i][j] = temp[j];
            }
        }
    }

    void DrawPesawat(){
        // drawing body of plane
        line((body[0][0]*sc + x), (body[0][1]*sc + y), (body[1][0]*sc + x), (body[1][1]*sc + y));
        line((body[1][0]*sc + x), (body[1][1]*sc + y), (body[2][0]*sc + x), (body[2][1]*sc + y));
        line((body[2][0]*sc + x), (body[2][1]*sc + y), (body[3][0]*sc + x), (body[3][1]*sc + y));
        line((body[3][0]*sc + x), (body[3][1]*sc + y), (body[4][0]*sc + x), (body[4][1]*sc + y));
        line((body[4][0]*sc + x), (body[4][1]*sc + y), (body[5][0]*sc + x), (body[5][1]*sc + y));
        line((body[5][0]*sc + x), (body[5][1]*sc + y), (body[6][0]*sc + x), (body[6][1]*sc + y));
        line((body[6][0]*sc + x), (body[6][1]*sc + y), (body[7][0]*sc + x), (body[7][1]*sc + y));
        line((body[7][0]*sc + x), (body[7][1]*sc + y), (body[8][0]*sc + x), (body[8][1]*sc + y));
        line((body[8][0]*sc + x), (body[8][1]*sc + y), (body[9][0]*sc + x), (body[9][1]*sc + y));
        line((body[9][0]*sc + x), (body[9][1]*sc + y), (body[10][0]*sc + x), (body[10][1]*sc + y));
        line((body[10][0]*sc + x), (body[10][1]*sc + y), (body[11][0]*sc + x), (body[11][1]*sc + y));
        line((body[11][0]*sc + x), (body[11][1]*sc + y), (body[12][0]*sc + x), (body[12][1]*sc + y));
        line((body[12][0]*sc + x), (body[12][1]*sc + y), (body[0][0]*sc + x), (body[0][1]*sc + y));

        // drawing wing of plane
        line((wing[0][0]*sc + x), (wing[0][1]*sc + y), (wing[1][0]*sc + x), (wing[1][1]*sc + y));
        line((wing[1][0]*sc + x), (wing[1][1]*sc + y), (wing[2][0]*sc + x
```

```
    ), (wing[2][1]*sc + y));
282         line((wing[2][0]*sc + x), (wing[2][1]*sc + y), (wing[3][0]*sc + x
    ), (wing[3][1]*sc + y));
283         line((wing[3][0]*sc + x), (wing[3][1]*sc + y), (wing[4][0]*sc + x
    ), (wing[4][1]*sc + y));
284         line((wing[4][0]*sc + x), (wing[4][1]*sc + y), (wing[5][0]*sc + x
    ), (wing[5][1]*sc + y));
285         line((wing[5][0]*sc + x), (wing[5][1]*sc + y), (wing[0][0]*sc + x
    ), (wing[0][1]*sc + y));
286     }
287
288     void saveBulletPos(){
289         xbullet = mousex();
290         ybullet = mousey();
291     }
292
293     void DrawBullet(){
294         circle(xbullet, ybullet, radbullet);
295     }
296
297     void moveBullet(){
298         ybullet = ybullet - stybullet;
299     }
300
301
302 };
303
304 class Bullet{
305 private:
306     float x;
307     float y;
308     float radius;
309     int Xmax ;  // screen x max
310     int Ymax;  // screen y max
311     int sty;  // determine bullet speed y
312 public:
313     Bullet(){
314         // x = mousex();
315         // y = mousey();
316         radius = 5;
317         Ymax = getmaxy();
318         Xmax = getmaxx();
319         sty = 30;
320     }
321
322     void saveBulletPos(){
323         x = mousex();
324         y = mousey();
325     }
326
327     void DrawBullet(){
328         circle(x, y, radius);
329     }
```

```
330
331     void moveBullet(){
332         y = y - sty;
333     }
334 };
335
336 int main(){
337     initwindow(500,500);
338     PesawatLakon PesawatKuh;
339     PesawatKuh.Skala(10);
340     Pesawat M[20];
341     // Bullet myBullet;
342     float xh,yh;
343     char c;
344     bool adaPeluru = false;
345
346     // harus dijalankan cuma sekali
347     // PesawatKuh.RotatePesawat(90);
348     for(int i=0; i<20; i++){
349         M[i].RotatePesawat(90);
350     }
351
352     do{
353         cleardevice();
354         if(kbhit()){  // keyboard hit
355             c = getch();  // get button of pressed keyboard
356             if (c == 27) break;
357         }
358
359         xh = mousex();
360         yh = mousey() ;
361         PesawatKuh.Posisi(xh, yh);
362         PesawatKuh.DrawPesawat();
363
364         // if(ismouseclick(WM_LBUTTONDOWN)){
365         //     printf("clicked once");
366         //     clearmouseclick(WM_LBUTTONDOWN);
367         // }
368
369         // nembak
370         if(ismouseclick(WM_LBUTTONDOWN)){
371             PesawatKuh.saveBulletPos();
372             clearmouseclick(WM_LBUTTONDOWN);
373         }
374         PesawatKuh.DrawBullet();
375         PesawatKuh.moveBullet();
376
377         // move pesawat ke kanan
378         for(int i=0; i<7; i++){
379             M[i].MoveRight();
380             M[i].DrawPesawat();
381
382             // for bullet
```

```
            if(!adaPeluru){
                printf("save bullet pos");
                M[i].saveBulletPos();
                adaPeluru = true;
            }
            M[i].DrawBullet();
            M[i].moveBullet();
            if(M[i].getYBullet() > getmaxy()){
                M[i].saveBulletPos();
            }
        }

        delay(50);  // delay 50ms each iteration
    } while(1);

    getch();
    closegraph();
    return 0;
}
```