 Template буюу загвар класс   
(Лаборатори №11)

ХШУИС – ПХ Х.Очирсүх 20B1NUM0429@stud.num.edu.mn

ХШУИС – КУ Н.Дагвадорж 20B1NUM2585@stud.num.edu.mn

ХШУИС – КУ Э.Рэнцэндондог 20B1NUM0054@stud.num.edu.mn

1. ОРШИЛ

Энэ лабораторийн ажлаар загвар функц, загвар класс, linked list гэж юу болох талаар багаараа судлан, C++ хэл дээр хэрхэн хэрэгжүүлдэг талаар дэлгэрэнгүй судална.

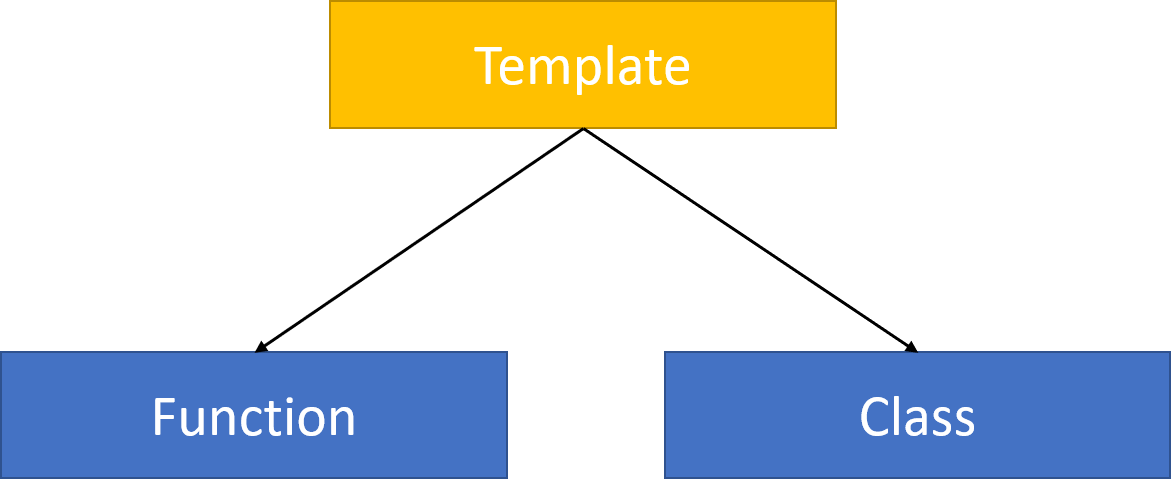
2. ЗОРИЛГО

Лабораторын ажил 7-г өргөтгөж дурын төрлийн өгөгдөл хадгалдаг жагсаалт зохион байгуулна.Үүний тулд дараах зорилтуудыг тавьж ажилласан:

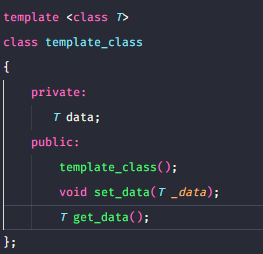
1. Загвар функц, загвар класс, жагсаалтын талаар судлах
2. Жагсаалт зохион байгуулах (linked\_list гэсэн класс үүсгэн түүний байгуулагчуудыг тодорхойлох)
3. Жагсаалтын сүүлд элемент нэмэх add(T, t) функц тодорхойлох
4. Жагсаалтын index – дүгээр байрлалд элемент нэмж оруулах insert(T t, int index) функц тодорхойлох
5. Жагсаалтын index – дүгээр элементийн утгыг буцаах get(int index) функц тодорхойлох
6. Жагсаалтын index – дүгээр элементийг устгах delete(int index) функц тодорхойлох
7. Жагсаалтын уртыг (элем-нтийн тоо) буцаах length() функц тодорхойлох
8. Үүсгэсэн жагсаалтаа загвар класс ашиглан дурын төрлийн өгөгдөлийг хадгалдаг болгон сайжруулах
9. Санамсаргүй талын урттай, санамсаргүй тооны (20-30 орчим) гурвалжин, дөрвөлжин, тойргийн объектуудыг үүсгэх
10. Объект эрэмбэлэх функц бичих
11. Функцуудыг тодорхойлохдоо дахин ашиглах зарчмыг баримтлах

3. ОНОЛЫН СУДАЛГАА

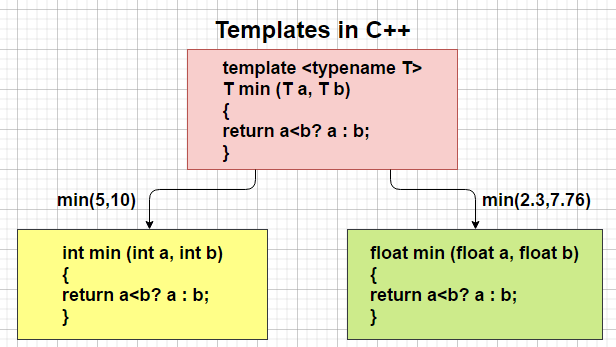
3.1 Загвар функц, загвар класс



**Загвар класс** (Template class). Ерөнхий (дурын) өгөгдлийн төрлүүдтэй ажиллах боломжийг олгодог ингэснээр төрөл тус бүр дээр дахин бичих шаардлагагүй.



**Загвар функц.** (Template function). Ерөнхий төрлүүдтэй харьцаж болох тусгай функц.



3.2 Linked List

Жагсаалт (linked list).

Жагсаалт нь шугаман дараалласан зохион байгуулалтай боловч элементүүд нь санах ойд дараалж байрладаггүй, санах ойд динамикаар байрладаг өгөгдлийн бүтэц юм.

Нэг холбоост жагсаалтын элемент буюу зангилаа нь өгөгдөл ба холбоос(заагч) 2 хэсгээс тогтдог. Заагч нь дараагийн элементийн хаягийг заана.

template<class T>

class linked\_list

{

public:

  struct  Elm

 {

    T  x;

    Elm \*next;

  };

typedef  struct Elm Elm;

    Elm \*head;

    Elm \*tail;

    int len;

public:

    linked\_list();

    void add(T x);

    void insert(T x, int idx);

    float get(int index);

    void delete\_(int idx);

    void pop\_front();

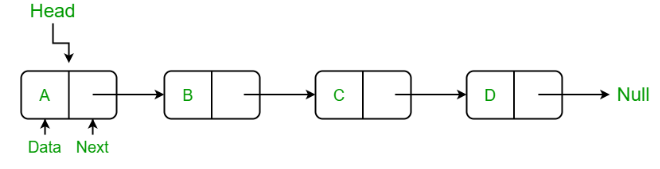
    void pop\_back();

    int length();

    void print();

    void swap(int, int);

};



3.3 rand function

Си програмчлалын хэлэнд rand() функц нь [0, RAND\_MAX] мужид санамсаргүй тоог үүсгэдэг бөгөөд stdlib.h толгой файлд тодорхойлогдсон байдаг.

int radius\_rand = rand() % 10 + 1;

4. ХЭРЭГЖҮҮЛЭЛТ

1. Зорилт, зорилтуудад бичсэн ажил, асуудлыг хэрхэн шийдсэн талаар бичнэ. Яг өөрийн хийсэн зүйлээ бичнэ. Кодоос хэсэгчлэн оруулж тайлбарлаж болно.

2. Кодыг ажиллуулсан туршилт, тэдгээрийн оролтын өгөгдөл, гаралтын өгөгдөл, алхам бүрийг тайлбайрлана. Гаралтын зургыг хавсаргаж болно.

Жагсаалтын сүүлд элемент нэмэх add(T, t) функц тодорхойлох

template<class T> void linked\_list<T>::add(T x){

      Elm\* o = new Elm;

      o->x = x;

      o->next = NULL;

      if(this->len == 0){

         this->head = this->tail = o;

         this->len++;

      }

      else{

         this->tail->next = o;

         this->tail = o;

         this->len++;

      }

}

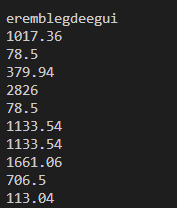
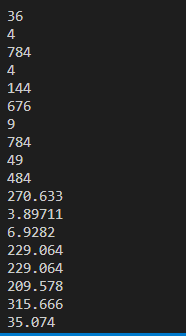
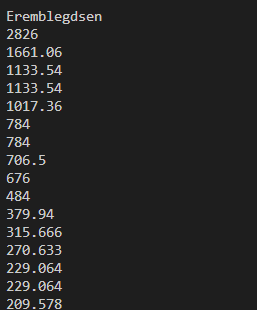
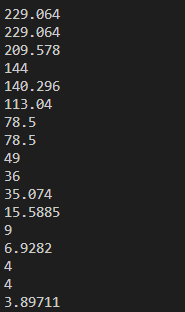
1. Жагсаалтын index – дүгээр байрлалд элемент нэмж оруулах insert(T t, int index) функц тодорхойлох
2. template<class T> void linked\_list<T>::insert(T x, int idx){
3. if(idx>=this->len) add(x);
4. else{
5. int i;
6. Elm \*o = new Elm;
7. Elm \* p1 = this->head;
8. o->x = x;
9. o->next = NULL;
10. this->len++;
11. for(i; i<idx-1; i++)
12. if(p1!=NULL);
13. p1 = p1->next;
15. o->next = p1->next;
16. p1->next = o;
18. }
19. }
20. Жагсаалтын index – дүгээр элементийн утгыг буцаах get(int index) функц тодорхойлох
21. template<class T> float linked\_list<T>::get(int idx){
22. Elm \*o = this->head;
23. int pos = 0;
24. for(o; o!=NULL; o=o->next){
25. if(pos == idx){
26. // cout<<"S: "<<o->x->s;
27. return o->x->s;
28. }
29. else pos++;
30. }
31. return -1;
32. }
33. Жагсаалтын index – дүгээр элементийг устгах delete(int index) функц тодорхойлох
34. template<class T> void linked\_list<T>::delete\_(int idx){
35. if(idx >= this->len|| idx < 0 || this->len == 0) return;
36. else if(idx ==0){
37. pop\_front();
38. }
39. else if(idx == this->len-1 && this->len != 0){
40. pop\_back();
41. }
42. else{
43. struct Elm \*p1=this->head;
44. struct Elm \*p2;
46. while(idx>1){
47. if(p1 != NULL)
48. p1 =p1->next;
49. idx--;
50. }
51. p2 = p1->next;
52. p1->next = p1->next->next;
53. free(p2);
54. this->len--;
55. }
56. }
57. Жагсаалтын уртыг (элем-нтийн тоо) буцаах length() функц тодорхойлох

template<class T> int linked\_list<T>::length(){

     return this->len;

}

Үр дүн :



1. ДҮГНЭЛТ

Энэхүү лаборатори 11 хүрээнд багаараа template класс болон функц, linked list ойлголтыг мэдэж авж, кодондоо хэрэгжүүлсэн. Template class болон функц ашигласнаар ерөнхий өгөгдлийн төрлүүдтэй ажиллах боломжыг олгодог. Ингэснээр функц дахин програмчлах шаардлагагүй болж, програмын алдааг бууруулж, програмын хэмжээг багасах давуу талыг үүсгэдэг. Мөн багаараа ажиллаж олон талын мэдээллийг цуглуулж, хоорондоо зөвлөлдөн, тулгарсан асуудлуудаа олон талын өнцгөөс харж шийдвэрлэж, бие биедээ мэдлэгүүдээ хуваалцсан. Үүний үр дүнд богино хугацаанд их асуудлуудыг шийдэж чадсан.

6. АШИГЛАСАН МАТЕРИАЛ

1. Объект хандлагат технологийн С++ програмчлал, Ж.Пүрэв, 2008, Улаанбаатар.

2. strcmp, <http://en.cppreference.com/w/c/string/byte/strcmp>

3. <https://www.geeksforgeeks.org/what-is-linked-list/>

4. <https://www.programiz.com/cpp-programming/class-templates>

5. Өгөгдлийн бүтэц ба Алгоритм Б.Отгонбаяр

7. ХАВСРАЛТ

List.h

#include<stdlib.h>

#include<stdio.h>

#ifndef \_list\_

#define \_list\_

template<class T>

class linked\_list

{

public:

  struct  Elm

 {

    T  x;

    Elm \*next;

  };

typedef  struct Elm Elm;

    Elm \*head;

    Elm \*tail;

    int len;

public:

    linked\_list();

    void add(T x);

    void insert(T x, int idx);

    float get(int index);

    void delete\_(int idx);

    void pop\_front();

    void pop\_back();

    int length();

    void print();

    void swap(int, int);

};

  // utgiig ni solih

template<class T> void linked\_list<T>::swap(int i, int j){

    Elm \*p1 =this->head;

    Elm \*p2 = this->head;

    int a;

    for(a=0; a<i; a++){

       if(p1!=NULL);

       p1 = p1->next;

    }

     for(a=0; a<j; a++){

       if(p2!=NULL);

       p2 = p2->next;

    }

     T sav;

     sav = p1->x;

     p1->x = p2->x;

     p2->x = sav;

}

template<class T> linked\_list<T>::linked\_list(){

        head = NULL;

        tail = NULL;

        len = 0;

    }

template<class T> void linked\_list<T>::add(T x){

      Elm\* o = new Elm;

      o->x = x;

      o->next = NULL;

      if(this->len == 0){

         this->head = this->tail = o;

         this->len++;

      }

      else{

         this->tail->next = o;

         this->tail = o;

         this->len++;

      }

}

template<class T> void linked\_list<T>::insert(T x, int idx){

    if(idx>=this->len) add(x);

     else{

        int i;

        Elm \*o = new Elm;

        Elm \* p1 = this->head;

        o->x = x;

        o->next = NULL;

        this->len++;

        for(i; i<idx-1; i++)

            if(p1!=NULL);

            p1 = p1->next;

       o->next = p1->next;

       p1->next = o;

     }

}

template<class T> float linked\_list<T>::get(int idx){

     Elm \*o = this->head;

     int pos = 0;

    for(o; o!=NULL; o=o->next){

         if(pos == idx){

       // cout<<"S: "<<o->x->s;

         return o->x->s;

         }

         else pos++;

    }

    return -1;

}

template<class T> void linked\_list<T>::pop\_front(){

    Elm \*o;

     if(this->head == NULL) return;

     else{

          o = this->head;

          this->head = this->head->next;

          delete o;

     }

}

template<class T> void linked\_list<T>::pop\_back(){

    Elm \*o = this->head;

    for(o; o->next != this->tail;)

    o = o->next;

    if(o == NULL){

        delete this->head;

        this->head = this->tail = NULL;

    }

    else{

         o->next = NULL;

         delete this->tail;

         this->tail = o;

         this->len--;

    }

}

template<class T> void linked\_list<T>::delete\_(int idx){

       if(idx >= this->len|| idx < 0 || this->len == 0) return;

   else if(idx ==0){

    pop\_front();

   }

   else if(idx == this->len-1 && this->len != 0){

    pop\_back();

   }

  else{

      struct Elm \*p1=this->head;

      struct Elm \*p2;

     while(idx>1){

      if(p1 != NULL)

      p1 =p1->next;

      idx--;

     }

      p2 = p1->next;

      p1->next = p1->next->next;

      free(p2);

      this->len--;

  }

}

template<class T> int linked\_list<T>::length(){

     return this->len;

}

template<class T> void linked\_list<T>::print(){

          Elm  \*o = this->head;

        while (o != NULL)

        {

          cout<<o->x->s<<endl;

          o = o->next;

        }

}

#endif

 Shape.h

#ifndef \_shape\_

#define \_shape\_

#include <iostream>

using namespace std;

class shape

{

protected:

    float a;

public:

    // void set\_a(float a);

    shape(float a);

    shape();

};

#endif

Shape.cpp

#include "shape.h"

shape::shape(){

}

shape::shape(float a)

{

      this->a = a;

}

Twod.h

#include "shape.h"

#ifndef \_twod\_

#define \_twod\_

class twod : public shape

{

public:

  float \*x;

  float \*y;

  float h, s, p;

public:

  virtual float findArea() = 0;

  virtual float findP() = 0;

  float get\_s();

  float get\_p();

  twod(float a, int points);

  twod();

};

#endif

Twod.cpp

#include "twod.h"

float twod::get\_s()

{

   return this->s;

}

float twod::get\_p(){

   return this->p;

}

twod::twod(float a, int points) : shape(a)

{

   this->a = a;

   this->x = new float[points];

   this->y = new float[points];

}

twod::twod():shape(){

}

Circle.h

#include "twod.h"

#ifndef \_cicrle\_

#define \_circle\_

#define pi 3.14

class circle : public twod

{

public:

  circle();

  circle(float a);

  float findArea();

  float findP();

  ~circle();

};

#endif

Circle.cpp

#include <iostream>

#include "circle.h"

circle::circle():twod(){

   cout << "\ncircle contructor vvsgev s= " << this->findArea() <<this->findP()<<"\n";

}

circle::circle(float a) : twod(a, 1)

{

   cout << "\ncircle contructor vvsgev s= " << this->findArea() <<this->findP()<<"\n";

}

float circle::findArea()

{

   this->s = this->a \* this->a \* pi;

   return this->s;

}

float circle::findP(){

    this->p = 2 \* this->a \*pi;

    return this->p;

}

circle::~circle()

{

   delete[] x;

   delete[] y;

}

Square.h

#include "twod.h"

#ifndef \_square\_

#define \_square\_

class square : public twod

{

public:

    float findArea();

    float findP();

    square(float a);

};

#endif

Square.cpp

#include "square.h"

float square::findArea()

{

   this->s = this->a \* this->a;

   return this->s;

}

float square::findP(){

   this->p = 4 \* this->a;

   return this->p;

}

square::square(float a) : twod(a, 4)

{

   cout << "\nsquare contructor vvsgev s: " << this->findArea() <<this->findP()<< "\n";

}

Triangle.h

#include "twod.h"

#ifndef \_triangle\_

#define \_triangle\_

class triangle : public twod

{

private:

    float h;

public:

    float findArea();

    float findP();

    void findX();

    void set\_h(float);

    float get\_h();

    triangle(float x);

};

#endif

Triangle.cpp

#include "triangle.h"

#include <math.h>

void triangle::findX()

{

   h = sqrt(pow(a, 2) - pow(a / 2, 2));

   set\_h(h);

}

void triangle::set\_h(float h)

{

   this->h = h;

}

float triangle::get\_h()

{

   return this->h;

}

float triangle::findArea()

{

   findX();

   this->s = (this->a \* get\_h()) / 2;

   return this->s;

}

float triangle::findP(){

   this->p = 3 \* this->a;

   return this->p;

}

triangle::triangle(float x) : twod(x, 3)

{

   cout << "triangle contructor vvsgev talbai s: " << this->findArea()<<this->findP() << "\n";

}

Main.cpp

#include <iostream>

using namespace std;

#include<stdlib.h>

#include "list.h"

#include "circle.h"

#include "square.h"

#include "triangle.h"

int main()

{

  int max = 10, min = 6, i, raduis\_rand, circle\_rand, j, square\_rand, triangle\_rand;

   circle\_rand = rand() % (max - min + 1) + min;

    twod \*shapes[30];

 cout<<"\nCircle sanamsargui too:"<<circle\_rand<<endl;

 for(i=0; i<10; i++){

    raduis\_rand = rand() % 30 + 1;

    shapes[i] = new circle(raduis\_rand);

 }

  for(i=10; i<20; i++){

    square\_rand = rand() % 30 + 1;

    shapes[i] = new square(square\_rand);

 }

  for(i=20; i<30; i++){

    triangle\_rand = rand() % 30 + 1;

    shapes[i] = new triangle(triangle\_rand);

 }

// list amjillttai heregjsen

    int ds, t, x, pos, s;

     linked\_list<twod\*> l;

     for(i=0; i<30; i++)

     l.add(shapes[i]);

     cout<<"\neremblegdeegui\n";

     l.print();

      for(i=30; i>0; i--){

       for(j=0; j<i; j++){

         if(l.get(i) > l.get(j)){

           l.swap(i,j);

         }

       }

    }

    cout<<"\nEremblegdsen\n";

    l.print();

   return 0;

}

template class linked\_list<twod \*>;