2-деңгей

1. Қай функцияның көмегімен жүйе графикалық режимге өтеді?

Дербес компьютер экраны текстік және графиктік режимдердің бірінде жұмыс істейді. Графикалық режимде жұмыстың қалай жүргізілетінін қарастырайық.

Қазіргі ДК-де, негізінен, растрлық дисплейлер қолданылады. Олардағы бейненің ең кіші элементі болып нүкте – **pixel** (ағылш., picture element) табылады. Дисплейдің мүмкіндігі – бұл көлденең және тік орналасқан пикселдер саны (стандартты разрешение – 640*480 нүкте). Дисплейде бейнеленетін сурет орталық процессор жадысының бейнежады (видеопамять) деп аталатын арнайы облысында кодталып сақталады. Мәліметтер периодты түрде осы облыстан оқылып, бейнесигналдарға түрлендіріледе экранда бейнеленеді. Сурет кодтарын бейнесигналдарға түрлендіруді арнайы электрондық схема-бейнеадаптер (видеоадаптер) жүзеге асырады. Ең кең тараған адаптерлер VGA және SVGA. C++ тілінде бейнежадының дербес бөліктеріне енуге, түстерді басқаруға, әр түрлі формада графикалық бейнелер тұрғызуға, текстік хабарламалар шығаруға, курсорды басқаруға мүмкіндік беретін көптеген функциялардын тұратын графикалық кітапхана бар. Бұл функциялардың нақты бейнеадаптерлермен жұмысқа баптау қажетті графикалық драйверді қосу арқылы қол жеткізіледі. Драйвер – бұл ДК-дің құрылғыларын басқаруға арналған арнайы программа. Адаптерлердің барлық түрлері үшін графикалық драйверлер Borland International фирмасымен құрастырылған. Олар BGI (Borland Graphics Interface) кеңеймесімен жеке файлдарда орналасқан. Графикалық драйверді қосу үшін арнайы initgraph() функциясы қолданылады.

Көптеген графикалық функциялар ағымдағы позиция көрсеткіші деген ұғымды қолданады. Ол таңдалған пикселді білдіреді және екі бүтін сан арқылы сипатталады: экранның горизонталь және вертикаль координаттары. Нумерация солдан оңға және жоғарыдан төмен қарай жүргізіледі (нольден бастап).

Графикалық жүйені инициализациялау.

1. Драйверді және графикалық режимді таңдау detectgraph (&gd, &gm) функция арқылы орындалады.
2. Драйверді жүктеу, графикалық жүйені инициализациялауды initgraph (&gd, &gm, "BGI-файлдың жолы") функциясы атқарады, мұндағы gd және gm айнымалылары қажет драйвер мен

графикалық режимнің номерлері. Егер BGI-файлдар ағымдағы директорияда орналасса, онда initgraph()функциясының үшінші параметрі ретінде бос жолды беруге болады initgraph (&gd, &gm, " ");

2. Координаталар нүктелерін бөлу қалай жүзеге асады?

Көптеген графикалық функциялар ағымдағы позиция көрсеткіші деген ұғымды қолданады. Ол таңдалған пикселді білдіреді және екі бүтін сан арқылы сипатталады: экранның горизонталь және вертикаль координаттары. Нумерация солдан оңға және жоғарыдан төмен қарай жүргізіледі (нольден бастап).

3. break, continue, goto операторлары не үшін қолданылады

C++ бағдарламалау тілінде 'break', 'continue' және 'goto' операторлары бағдарламаның орындалуын басқару және циклдар мен шарттылықтардың ішіндегі басқару ағынын өзгерту үшін қолданылады. Олардың әрқайсысын толығырақ қарастырайық:

1. **`break`**: `break` операторы циклден (мысалы, `for`, `while`, `do-while`) немесе `switch` блогынан аяқталу үшін пайдаланылады. цикл мерзімінен бұрын немесе қосқыш конструкциялардан шығу. Мысалы:
2. **` continue`**: «continue» операторы ағымдағы цикл итерациясының қалған бөлігін өткізіп жіберу және келесі итерацияға өту үшін пайдаланылады. Мысалы:
3. **`goto`**: `goto` операторы бағдарламаның белгілі бір нүктесіне өтуге мүмкіндік беретін белгі болып табылады. Дегенмен, «goto» пайдалану жақсы тәжірибе болып саналмайды, себебі ол кодты түсінуді және жөндеуді қиындатады. Оның орнына, әдетте орындалу ағынын басқару үшін циклдар мен шарттар сияқты деректер құрылымдарын пайдалану ұсынылады.

Айта кету керек, «goto» пайдалану ұсынылмайды және көп жағдайда құрылымдық бағдарламалау конструкцияларын пайдалану арқылы оны болдырмауға болады.

4. Массив дегеніміз не, оның жазылу түрін көрсетіңіз

Массив дегеніміз — бір атпен берілген айнымалылар тізбегі. Массивтің аты және өлшемі болады. Си тілінде массивті былай сипаттаймыз:

Айнымалылыр типі массив_аты [элементтер саны]

Мысалы: char irr [10], float git [5]

Си программалау тіліндегі қабылдайтын мән 0-ден бастап n-1 аралығына дейінгі мәндерді қабылдайды.

Массив элементтерін енгізуді ұйымдастыру.

Мыс1: A[10] массив элементін енгіз.

1. Перне тақтаның көмегімен енгізу.

```
#include <stdio.h>
#include <stlib.h>
#include <conio.h>
void  main()
{ const int N=10;
  int A[N];I;
  for (i=1;i<N;i++)
  {
    printf  ( “\nA[%d]→”,i);
    scanf (“%d”,&A[i]);
}
```

5. Массивтің түрлері қандай және қай кезде қолданылады?

C++ тілінде массивтердің бірнеше түрі бар, олардың әрқайсысының өзіндік сипаттамалары бар. Міне, массивтердің кейбір түрлері:

1. **Статикалық массив:**
   - Компиляция уақытында белгіленген өлшеммен жарияланған.
   - Бағдарлама жұмыс істеп тұрғанда массив өлшемі өзгермейді.
   - Декларация: `int myArray[5];`
2. **Динамикалық массив:**
   - Өлшемді бағдарламаны орындау кезінде өзгертуге болады.
   - `new` операторы арқылы жасалған.
   - `delete` операторы арқылы жадты босату қажет.

Әрбір массив түрінің өзіндік артықшылықтары мен кемшіліктері бар және таңдау нақты тапсырмаға байланысты. Динамикалық массивтер мен векторлар бағдарламаны орындау кезінде массив өлшемі өзгеруі мүмкін жағдайлар үшін қолайлы. Vector сонымен қатар деректермен жұмыс істеудің көптеген ыңғайлы әдістерін ұсынады. Стектер мен кезектер функция шақыруларын басқару немесе деректерді алынған ретпен өңдеу сияқты белгілі бір тапсырмалар үшін пайдалы.

6. Массивті қалай программада қалай сипаттайды

```
// Статический массив
int staticArray[5]; // массив из 5 элементов типа int
// Динамический массив
int* dynamicArray = new int[5]; // динамический массив из 5 элементов типа int
// Вектор (часть библиотеки STL)
#include <vector>
std::vector<int> myVector(5); // вектор из 5 элементов типа int
```

7. Бір өлшемді массив дегеніміз не, мысал келтір

Массив дегеніміз — бір атпен берілген айнымалылар тізбегі. Массивтің аты және өлшемі болады. Си тілінде массивті былай сипаттаймыз:

***Айнымалылыр типі массив_аты [элементтер саны*]**

Мысалы: char irr [10], float git [5]

```
int myArray[5] = {1, 2, 3, 4, 5};
```

Бір өлшемді массив (немесе жай ғана «массив») – жадта ретімен орналасқан бір типті элементтердің реттелген жиыны. Әрбір массив элементінің әдетте 0-ден басталатын өз индексі болады.

8. Функция дегеніміз не, мысал келтір+

Бағдарламалауда функция белгілі бір тапсырманы орындайтын код бөлігі болып табылады. Функциялар кодты құрылымдауға, кодтың қайталануын болдырмауға және бағдарламаңызды оқуға және техникалық қызмет көрсетуге мүмкіндік береді. C++ тілінде функцияларды келесідей жариялауға және анықтауға болады:

```cpp
#include <iostream>
// Пример объявления функции
int add(int a, int b);
int main() {
    // Пример вызова функции
    int result = add(3, 4);
    // Вывод результата
    std::cout << "Результат сложения: " << result << std::endl;
    return 0;
}
// Пример определения функции
int add(int a, int b) {
    return a + b;
}
```

В этом примере:

- Функция `add` объявлена перед функцией `main` с использованием заголовка `int add(int a, int b);`. Этот заголовок сообщает компилятору о том, что существует функция с именем `add`, которая принимает два аргумента типа `int` и возвращает значение типа `int`.

- В функции `main` вызывается функция `add(3, 4)`, передавая ей аргументы 3 и 4. Результат сложения сохраняется в переменной `result`.

- Наконец, функция `add` определена после функции `main`. В теле функции указано, что функция принимает два аргумента `a` и `b`, и возвращает их сумму.

9. Функцияны анықтау деген не және оның синтаксисі+

10. Функцияны шақыру деген не және оның синтаксисі+

11. Фактілі аргумент деп нені айтады

Нақты аргумент (немесе жай ғана «аргумент») функция шақырылған кезде оған берілетін мән болып табылады. Функция өз денесінде әрекеттерді орындау үшін нақты аргументтерді пайдаланады.

Бағдарламалау контекстінде функцияны шақырған кезде функцияға нақты аргументтерді бересіз. Бұл аргументтер функция ішінде жергілікті айнымалылар сияқты пайдаланылады.

C++ тіліндегі мысал:

```
#include <iostream>

// Екі нақты аргументі бар функцияның мысалы
void printSum(int a, int b) {
    int сомасы = a + b;
    std::cout << "" << a << " және " << b << " қосындысы мынаған тең: " << sum << std::endl;
}

int main() {
    // Нақты аргументтерді беретін функцияны шақырыңыз
    printSum(3, 4);

    қайтару 0;
}
```

Бұл мысалда "printSum" функциясы "int" түріндегі екі нақты аргументті қабылдайды. Функцияны шақырған кезде '3' және '4' аргументтері 'негізгі' параметріне жіберіледі. Функцияның ішінде бұл мәндер соманы есептеу үшін пайдаланылады, содан кейін ол экранда көрсетіледі.

Сонымен, нақты аргументтер - бұл функцияға оның тапсырмаларын орындау үшін онымен жұмыс істеуі үшін берілетін деректер.

12. Рекурсивті шақыруға түсініктеме бер

Бағдарламалаудағы рекурсивті шақыру функцияны өзінен шақыруды білдіреді. Бұл әдіс көбінесе төменгі деңгейдегі ішкі мәселелерге табиғи түрде бөлінген мәселелерді шешу үшін қолданылады. Рекурсияны қолданатын функция рекурсивті функция деп аталады.

Рекурсивті функцияның негізгі элементтері:

1. **Негізгі жағдай:** Бұл рекурсия аяқталатын және функция енді өзін шақырмайтын шарт. Негізгі жағдай рекурсивті қоңыраулардың шексіз жұмыс істеуіне жол бермейді.

2. **Рекурсивті қадам:** Бұл рекурсивті шақыру орын алатын және тапсырма кішірек ішкі тапсырмаларға бөлінген функцияның бөлігі.

Санның факториалын есептеу үшін C++ тіліндегі рекурсивті функцияның мысалы:

```
#include <iostream>

// Рекурсивная функция для вычисления факториала
unsigned long long factorial(int n) {
    // Базовый случай: факториал 0 равен 1
    if (n == 0 || n == 1) {
        return 1;
    } else {
        // Шаг рекурсии: n! = n * (n-1)!
        return n * factorial(n - 1);
    }
}

int main() {
    // Пример вызова рекурсивной функции
    int num = 5;
    unsigned long long result = factorial(num);

    std::cout << "Факториал " << num << " равен: " << result << std::endl;

    return 0;
}```
```

Бұл мысалда «факторлық» рекурсивті функция болып табылады. Негізгі жағдай `n == 0` немесе `n == 1` мәніне орнатылады және рекурсия қадамы `(n-1)` факториалы арқылы `n` факториалын есептейді. Негізгі жағдайға жеткенде, рекурсия аяқталады.

Рекурсивті функциялар мәселені неғұрлым табиғи және қысқаша сипаттау үшін жиі пайдаланылады, оларды қарапайым және түсінікті ішкі тапсырмаларға бөледі. Дегенмен, тым терең рекурсия кезінде қоңыраулар стекінің толып кетуін болдырмау үшін рекурсияны пайдалану кезінде абай болу маңызды.

13. Құрылым деген не, мысал келтір

Құрылым деп- әртүрлі немесе бір типті айнымалыларды бір атпен бір тілдік құрылыммен топтастыру. Құрылыммен біріктірілген айнымалылар мүшелер, элементтер немесе құрылымның өрісі деп аталады. Құрылымды (немесе құрылымдық шаблондарды) хабарлауды struct сөзі пайдаланады., одан кейін идентификатор, құрылым типінің аты, тізімдер өрістері немесе құрылымның мүшелері фигуралық жақшаға алынған. Құрылым жариялайтын әрбір мүше, құрылымның өзі нүкте үтірмен аяқталады:

```
struct date    {
int day;
int month;
};
```

Құрылымды пайдалану мүмкіндігін алу үшін, алдымен құрылымдық типті айнымалыны жариялауымыз керек. Оны екі тәсілмен жасауға болады:

1) **struct date today;**
2) **struct date {int day; int month; } today;**

Екінші тәсіл бір уақытта құрылымды жариялап және құрылымдық типті айнымалыға жадыда орын бөледі. Бір құрылымдыдағы бірнеше құрылымдық айнымалыларды жариялауда, оларды үтірмен айырып жазу керек: **struct date dl,d2,d3;**

Құрылымды жариялауда typedef қызметші сөзі қолданады, ол берілгендер типтерінің бүркеншек атын құрайды: **typedef псевдонимді жариялау;**

Мысалы: Addr құрылымы сипатталған, адрестің 4 компоненті бар: қала, көше, үй, пәтер.

```
#include <stdio.h>
void main() {
typedef  struct addr {
char town [10];
char street [10];
int block;
int flat;
} addr;
```

```
Addr Home;
/* құрылым мүшелерінің мәндерін меншіктеу орындалады*/
Home.town = “Taraz”;
Home.street = “Alatau”;
Home.block = 90;
Home.flat = 12;
printf ("The address %s%s%04d%04d \n", Home.town,Home.street,Home.block,Home.flat);
      // берілгендерді массивке жазамыз
 }
```

Массив деп - бірдей түр элементтерінің жиынтығын атайды. Құрылым жиынтық элементінің шығу түрі болып табылады.

14. Бағдарламада құрылым қалай сипатталады+

15. Жазба мүшелері деп нені айтады ??????????

16. Ағымдар дегеніміз не, мысал келтір

Си++ енгізу-шығару ағымы. Си++ тілінде программалау кезінде Си тілінің stdio.h тақырыптық файлымен қосылатын стандартты енгізу-шығару жабдықтарын пайдалануға болады. Бірақ Си++ өзінің ерекше енгізу-шығару жабдықтары бар. Бұл iostream.h файлының көмегімен қосылатын кластар кітапханасы. Бұл кітапханада объектілер ретінде стандартты символдық ағымдар анықталған:

cin — пернетақтадан стандартты енгізу ағымы;

cout — экранға стандартты шығару ағымы.

Мысалы, x айнымалыға мәнді енгізу, мына оператормен іске асырылады.

cin>>x;

cout ағымында тырнақшаға алынған мәтіндер, өрнектер мәні шығарылады. Ағымға орналасудың операция белгісі <<. Ағымды шығарудың мысалдары:

17. Файл дегеніміз не, мысал келтір

Файлдық типті айнымалыларды сипаттау. Файлды қолдану режимдері

Файлдан оқу және файлға енгізу үшін ең алдымен файл fopen функциясының көмегімен ашылуы тиіс. Бұл функция операциялық жүйе арқылы орындалатын әрекеттерді ұйымдастыру жұмысын орындайды және файлмен мәлімет алмасуға арналған **көрсеткішті** қайтарады.

Ал, файлға көрсеткіш файл туралы информациялардан тұратын құрылымға (жазбаға) сілтейді. Мұндағы информация мынадай сұрақтарының жауабынан тұрады:

- буфер адресі,
- буфердегі ағымды литердің күйі,
- файлдан оқуға немесе жазуға ашық па?,
- файлдың соңғы таңбасы кездесті ме?

18. Файлды ашу функциялары, атап көрсетіңіз

Мұндай құрылым сипаттамасы **<stdio.h>** кітапханасындағы FILE типіндеберіледі. Қолдану үшін мынадай декларация берілсе жеткілікті:

FILE * fp;

FILE * fopen (char * name, char * m);

Мұндағы: fp – FILE типіндегі көрсеткіш, ал fopen FILE – ге көрсеткішті қайтарады. Fopen функциясы мына түрде қолданылады:

fp = fopen (name, m);

Мұндағы: name – файлдың атын меншіктейтін жол; ал, **m** – файлды қолдану режимі; яғни бұл да жол, қолданушы файлды қалай қолданатынын білдіреді, төмендегідей мәндердің бірін иелене алады:

" **r** " – (read) оқу режимі;

" **w** "- (write) жазу;

" **a** "- (apprnd) толықтыру;

Кейбір жүйеде текстік және бинарлық файлдар болып жіктеледі, бұл жағдайда режим жолына "**в**" (binary – бинарлық) немесе “t”(текстік) таңбасы тіркеледі. Файлмен жұмыс жасау барысында қате кездессе, онда fopen функциясы NULL мәнін қайтарады.

Мысалы, FILE*fin, * fout;

fin=fopen (“PRIMER.dat”,”’r”);

fout=fopen (“RESULT.dat”, “w”);

1-ші жолда 2:fin,fout – файл көрсеткіші құрылады, ал төменгі жолдарда сәйкесінше оқуға және жазуға арналған файлдар ашылады.

Fopen() функциясы файлдың аталған көрсеткішін қабылдайды, программаның орындалу барысында олардың мәндері жасанды өзгертілмеуі тиіс. Мұндай файлдағы 2-ші параметр файлдарымен мәлімет алмасу режимін анықтайды. Файл мынадай режимдерде ашылуы мүмкін:

- текстік;
- екілік;

Текстік режимде ашу үшін режимді көрсеткенде қасына «t» символын тіркеу арқылы жүзеге асады;

19. Текстік файлдар және қолдану жолдарын айтып бер

Мәтіндік файлдар ақпаратты мәтіндік формада сақтайтын файлдар. Олар адам немесе бағдарлама оқи алатын және түсіндіре алатын таңбалар тізбегі. Мәтіндік файлдардың кейбір негізгі сипаттамалары және оларды пайдалану жолы:

1. **Мәтіндік файл пішімдері:**
   - **TXT:** Пішімдеусіз қарапайым мәтінді қамтитын қарапайым мәтін пішімі.
   - **CSV (үтірмен бөлінген мәндер):** Мәндер үтірмен бөлінген мәтін пішімі. Кестелік деректерді сақтау үшін қолданылады.
   - **JSON (JavaScript Object Notation):** Кілт-мән жұптары түрінде деректерді беру үшін пайдаланылатын мәтін пішімі. Веб-бағдарламалауда жиі қолданылады.
   - **XML (eXtensible Markup Language):** Бағдарламалар арасында құрылымдық деректерді алмасуға арналған белгілеулері бар мәтін пішімі.

2. **Мәтіндік файлдарды пайдалану:**
   - **Деректерді сақтау:** Мәтіндік файлдарды мәтіндік жазбалар, конфигурация файлдары, бағдарламаның бастапқы коды және т.б. сияқты әртүрлі деректерді сақтау үшін пайдалануға болады.
   - **Деректерді алмасу:** Олар көбінесе бағдарламалар арасында деректер алмасу үшін қолданылады. JSON және XML пішімдері құрылымдық деректерді беру үшін әсіресе пайдалы.
   - **Мәтінді өңдеу:** Мәтіндік файлдар әртүрлі бағдарламалау тілдерін пайдаланып мәтіндік ақпаратты өңдеу және талдау кезінде жиі пайдаланылады.
   - **Конфигурация файлдары:** Мәтіндік файлдарды өңдеуді жеңілдететін бағдарлама параметрлері мен конфигурацияларын сақтау үшін пайдалануға болады.

Пример открытия текстового файла для записи:

```
#include <fstream>
#include <iostream>

int main() {
    // Открытие файла для записи
    std::ofstream outputFile("output.txt");

    // Проверка, открыт ли файл успешно
    if (!outputFile.is_open()) {
        std::cerr << "Не удалось открыть файл!" << std::endl;
        return 1; // Возвращаем код ошибки
    }

    // Запись данных в файл
    outputFile << "Пример записи в файл." << std::endl;

    // Закрытие файла
    outputFile.close();

    return 0;
}
```

20. Файлдармен жұмыс кезіндегі функцияларға сипаттама бер

C++ тілінде файлдармен жұмыс істеу стандартты `<fstream>` кітапханасының көмегімен жүзеге асырылады. Мұнда C++ тілінде файлдармен жұмыс істеуге арналған негізгі функциялар берілген:

1. **Файлды ашу:**
   - `std::ifstream` (файлды енгізу) және `std::ofstream` (файл шығысы) сәйкесінше оқу және жазу үшін файлды ашу үшін пайдаланылады.
   - `std::fstream` оқу және жазу үшін файлды ашуға мүмкіндік береді.

```cpp
```

```
#include <fstream>

std::ifstream inputFile("input.txt");
std::ofstream outputFile("output.txt");
std::fstream файлы("data.txt", std::ios::in | std::ios::out);
```

2. **Файлдың ашылуын тексеру:**
   - Файлды ашқаннан кейін оның сәтті ашылғанын тексеру керек.

```cpp
егер (inputFile.is_open()) {
    // Файл оқу үшін сәтті ашылды
} басқа {
    // Файлды ашу қатесін өңдеу
}
```

3. **Файлдан оқу:**
   - Файлдан деректерді оқу үшін `>>` операторын пайдаланыңыз.

```cpp
int мәні;
inputFile >> мәні;
```

4. **Файлға жазу:**
   - Файлға деректерді жазу үшін `<<` операторын пайдаланыңыз.

```cpp
outputFile << «Сәлеметсіз бе, файл!»;
```

5. **Файлды тексерудің соңы:**
   - `eof()` функциясын оқу кезінде файлдың соңына жеткенін тексеру үшін пайдалануға болады.

```cpp
while (!inputFile.eof()) {
    // Файлдан деректерді оқу
}
```

6. **Файлды жабу:**
   - Жұмысты аяқтағаннан кейін файлды жабу үшін `close()` әдісін пайдаланыңыз.

```cpp
inputFile.close();
outputFile.close();
```

7. **Файлдағы орналасу:**

- `seekg()` және `seekp()` сәйкесінше оқу және жазу үшін файл орнының көрсеткішін жылжыту үшін пайдаланылады.

```cpp
file.seekg(0, std::ios::beg); // Файлдың басына жылжытыңыз
```

8. **Оқылғаннан кейін файлдың соңын тексеру:**
   - Оқу/жазу операцияларының сәттілігін тексеру үшін `fail()` немесе `good()` қолдануға болады.

```cpp
егер (inputFile.fail()) {
    // Файлдан оқу қатесін өңдеңіз
}
```

Бұл функциялар C++ тілінде файлдармен жұмыс істеудің негізгі құралдарын қамтамасыз етеді. Қателерді өңдеу және пайдаланудан кейін файлдарды жабуды қарастыру маңызды.

21. Файл режимдерін ата, мысал келтір

Файл мынадай режимдерде ашылуы мүмкін:
- текстік;
- екілік;

Текстік режимде ашу үшін режимді көрсеткенде қасына «t» символын тіркеу арқылы жүзеге асады;

Режимдер кестесі:

им	аты
	л мәндерімен толықтырылуы үшін ашылады. Егер файл болса, ол құрылады. Жаңа мән соңына тіркеледі; арыдағы секілді, тек оқуға да болады; лды тек оқу үшін ашады, файл жоқ болса, ашылмайды; лды оқуға да, жазуға да болады; а файл ашады, бұрын бар болса, мәнін өшіреді. а файлды мән жазуға да, одан оқуға да мүмкіндік еді. Файл болса, мәні тазартылады.

R+,w+,a+ режимдерін қолдануда, яғни оған оқуды және жазуды бір уақытта орындалу барысында файл көрсеткішінің Fsetpos(), fseek(), немесе rewind()файлдары көмегімен ағымды позицияларын модификациялау керек. Программа жұмысы аяқталғанда Си-де автоматты түрде барлық ашық файлдар жабылады, ал файлды жабу үшін fclose(fin) қолданылады.

22. Графикалық режимді қалай орнатуға болады

Графикалық режимді тағайындау арнайы құралдардың көмегімен жүзеге асырылады. C++ тілінде графикалық құралдардың көпшілігі Паскаль тіліндегі графикалық құралдарға ұқсас. C++ тілінде де графикалық режимді тағайындау үшін initgraph(&gdriver,&gmode, PATHDRIVER); функциясы пайдалаиылады.

Мұндағы gdriver видео жүйе драйверін анықтайды, ал gmode видео жүйенің жұмыс режимін орнатады, PATHDRIVER драйвер файлының орнын қайда орналасатынын көрсетеді. Оны тұрақтылар бөлімінде төмендегідей етіп сипаттаймыз:

#defme PATHDRIVER "C:\YTCWbgi"

DRIVER шамасы әрдайым DETECT мәнін қабылдайды.

Графикалық режимнің жұмысын аяктап, оны жабу үшін, оған жадыдан берілген орынды босату үшін Closegraph() функциясы пайдаланылады.

Жалпы алғанда, компьютерде негізгі екі экран режимінің жұмысы - символдық және графикалық экран режимдері пайдаланылады. Компьютерді қосып, С++ жүйесін шақырганда мәтіндік режимде жұмыс істейді. Графикалық режимді алу үшін *graphics. h* функциясын, керекті графикалық режим *initgraph* процедурасымен инициализация жасалуы кажет. Режимді инициализациялау дегеніміз - дисплей адаптерінің жұмысын берілген графикалық режимнің күйіне келтіру, яғни физикалық экранды осы режимнің жұмысына көшіру.

23. Графикалық режимдегі қолданылатын функцияларды ата

Graphics.h функциясы

Графикалық режимді тағайындағаннан кейін пайдаланушы өзінің жұмысы үшін әртүрлі команда пайдаланады. Бұл командалар қызметі бойынша төмендегідей топталады:

Графикалық экранды басқару үшін;

- графикалық информацияны өңдеу және шығару үшін;
- графикалық режимде мәтін шығару үшін.

Графикалық экранды басқару командалары:

- пайдаланылған адаптер мен драйвер туралы информация алу;
- графикалық режимдердің мөлшерін және сипаттамаларын білу;
- графикалық экранның жұмысына қажетті режимді тағайындау;
- экранның графикалық беттерін басқару;
- графикалық информациямен шығаруға арналған терезені іске қосу және ажырату;
- графикалық информациямен шығару және фон түстерін басқару;
- экранды графикалық курсормен басқару;
- экранды немесе терезені тазарту жатады.

Графикалық информациямен енгізу және шығару командалары:

- графикалық экранға нүкте, кесінді, тіктөртбұрыш, қисық сызық, шеңбер, эллипс, доға, эллипс және шеңбер секторларын шығару процедуралары;
- динамикалық жадыға графикалық экранның бөліктерін бейнелерімен сақтау және оны қайтадан экранға шығару;
- экрандағы тұйықтарды берілген түстермен бояу немесе штрихтау.

Графикалық режимде экранға мәтін шығару командалары - графикалық экранға зігіпц типті мәтін жолын шығаруға мүмкіндік береді.

GRAPH модулінің командалары графикалық экранда кез-келген бейнені тұрғызуга мүмкіндік береді. Графикалық экранды басқарудың негізгі командалары

Бұл командалардың кейбіреуі жоғарыда айтылды.

- initgraph- графикалық режимді тағайындау;
- DETECT - графикалық драйверді автоматты түрде анықтау;
- closegraph() - графикалық режимді жабу;
- restorecrtmode() - графикалық режимнен уақытша мәтіндік режимге ауысу;
- setgraphmode(gm) - графикалық режимге қайтып оралу:

getmaxx() және getmax() функциясі - тағайындалған графикалық режимде экранның x және у осьтері бойынша ең үлкен координатасының мәнін анықтайды. Типі *integer* форматы: getmaxx(), getmaxy().

Графикалық курсорды басқару get(x) және get(y) функциялары курсордың ағымдағы координатасын анықтайды. Форматы: get(x) және get(y)

Фон түсін тагайындау және оны ғрафикалық экранға шыгару

Фонның түстерін басқару және шығару мүмкіндіктері графикалық экранға әртүрлі графикалық және мәтіндік информацияны түрлі түспен шығаруға мүмкіндік береді. Ол үшін кезекті бейнені шығарар алдында тағайындалған фонның түсін өзгерту жеткілікті. Сонымен қатар, түсті басқару командасы кез келген шығарылған графикалық нүктенің түсін анықтауға мүмкіндік береді; бұл графикалық информацияларды өңдеу алгоритмдерінде жиі пайдаланылады.

setcolor процедурасы - графикалық немесе мәтіндік информацияны графикалық экранға шығаратын ағымдағы түсті тағайындайды:

setcolor (<түс>).

Мұндағы <түс> : : *Word* - түс номерін көрсетеді. Мысалы, Setcolor(RED)немесе setcoIor(4);

Setbkcolor процедурасы орындалғанда графикалық экранның ағымдағы фонының түсі берілген түске өзгереді. Форматы:

Setbkcolor (<түсі>)Мұндағы түс WORD типті түстің номері.

Мысалы, setbkcolor(GREEN); немесе setbkcolor(6);

Графикалық информацияны шығару командалары

Графикалық экранға шығарудағы негізгі жасалатын әрекет графикалық нүкте шығару операциясы болып табылады. Экрандағы кез-келген бейне керекті туске боялған графикалық нүктелердің жиынтығынан құралады.

Графикалық экранға жиі пайдаланылатын графикалық объектілерді шығаруды жеңілдету үшін кесінді, сынық, тіктөртбұрыш, шеңбер, эллипс, доға, щеңбер және эллипс секторларын тұрғызатын процедуралар GRAPH модулінде қолданылады. Бұл процедуралар пайдаланушының осы объектілерді тұрғызуды бағдарламалаудан босатады (құтқарады). Бірақ модульде бір ғана графикалық нүктенің экрандағы еркін бейнесін шығаратын PutPixel процедурасы бар.

Графикалық экранға нүкте шығару

PutPixel процедурасы экранға нүктені көрсетілген координатасымен және түсімен шығарады.

Форматы: Puitpixel(x,y, C); мұндағы x, y -типі int, нүктенің координатасы, C++ - түсі.

Графикалық экранға кесінді салу

Line процедурасы бастапқы және соңғы нүктелерінің көрсетілген координатасы бойынша кесінді салады.

Форматы Line(x1, y1, x2, y2); мұндағы x1, y1 - бастапқы іiүктенің координатасы, x2, y2 - соңғы нүктенің координатасы.

Linerel процедурасы - кесіндіні ағымдағы түстен курсор тұрған нүктеден қосымша өзгертілген нүктенің координатасына дейін кесінді сызады.

Форматы:Linerel(dx, dy). Мұндағы dx, dy - қосымша өзгертілген (өсімше) нүктенің координатасы.

Кесіндіні шығару стилі

/сызықтың қалындықтары және түрлері/

Setlinestyle функциясы шығаратын сызықтың түрін тағайындайды. Ол тек кесінді үшін ғана емес, сондай-ақ кез келген геометриялық объектілерді сызу үшін де пайдаланылады. Модульге шығарылатын кесіндінің стандартты стильдерінің коды 0..3 енгізілген, ал 4-код бойынша пайдаланушының анықтайтын стилі алынады. Олар модульдің белгіленген тұрақтылары болып табылады:

SolidLn=0; (_____ тұтас)

DottedLn=l; (..... нүктелк)

SenterLn-2; (_._._._пунктирлі-штрих)

DashedLn=3; (---------- пунктир)

UserBitln=4; (пайдаланушы аныктайды) Пайдаланушының стилі 16 пиксельдің қатарынан жануын кодтауға мүмкіндік береді. Бұдан 16 биттен тұратын жол шығады: 1 - пиксельдің ағымдағы түспен жанатынын көрсетеді; ал 0- керісінше.

Setlinestyle процедурасы экранға шығарылатын кесінділердің стилін тағайындайды.

Форматы: Setlinestyle (<код>, <шаблон>, <қалыңдық>) Мұндағы <код>: - мәні 0 мен 4 аралығындағы стиль коды, <шаблон>: - пайдаланушы стилінің шаблонының коды, ол стиль коды тек 4-ке тең болған жағдайда пайдаланады.

24. Сызықты объектілерді салу функциялары

Графикалық экранға кесінді салу

Line процедурасы бастапқы және соңғы нүктелерінің көрсетілген координатасы бойынша кесінді салады.

Форматы Line(x1, y1, x2, y2); мұндағы x1, y1 - бастапқы ііүктенің координатасы, x2, y2 - соңғы нүктенің координатасы.

Linerel процедурасы - кесіндіні ағымдағы түстен курсор тұрған нүктеден қосымша өзгертілген нүктенің координатасына дейін кесінді сызады.

Форматы:Linerel(dx, dy). Мұндағы dx, dy - қосымша өзгертілген (өсімше) нүктенің координатасы.

Кесіндіні шығару стилі

/сызықтың қалындықтары және түрлері/

Setlinestyle функциясы шығаратын сызықтың түрін тағайындайды. Ол тек кесінді үшін ғана емес, сондай-ақ кез келген геометриялық объектілерді сызу үшін де пайдаланылады. Модульге шығарылатын кесіндінің стандартты стильдерінің коды 0..3 енгізілген, ал 4-код бойынша пайдаланушының анықтайтын стилі алынады. Олар модульдің белгіленген тұрақтылары болып табылады:

SolidLn=0; (_____ тұтас)

DottedLn=l; (..... нүктелк)

SenterLn-2; (_._._._пунктирлі-штрих)

DashedLn=3; (---------- пунктир)

UserBitln=4; (пайдаланушы аныктайды) Пайдаланушының стилі 16 пиксельдің қатарынан жануын кодтауға мүмкіндік береді. Бұдан 16 биттен тұратын жол шығады: 1 - пиксельдің ағымдағы түспен жанатынын көрсетеді; ал 0- керісінше.

Setlinestyle процедурасы экранға шығарылатын кесінділердің стилін тағайындайды.

Форматы: Setlinestyle (<код>, <шаблон>, <қалыңдық>) Мұндағы <код>: - мәні 0 мен 4 аралығындағы стиль коды, <шаблон>: - пайдаланушы стилінің шаблонының коды, ол стиль коды тек 4-ке тең болған жағдайда пайдаланады.

Экранда көпбұрыштар тұрғызу

Rectangle процедурасы диагональдарының төбелерінің координатасы бойынша тіктөртбұрші сызады.

Форматы:*Rectangle* (x1, y2, x2, y2); мұндағы x1, y2 – тіктөртбұрыштың сол жақ жоғарғы бұрышының координатасы, ал x2, y2 - оң жак төменгі бұрышының координатасы.

Bar процедурасы - ағымдагы түспен боялған тіктөртбұрыш сызады.

Форматы: *Bar* (x1, y 1, x2, y2);

(X1,УI) және (X2,У2) нүктелері боялған тіктөртбұрыштың сол жак жоғаргы және оң жақ төменгі бұрыштарының нүктелерінің координаталары. setfillstyle функциясының көмегімен бояудың түсі және үлгісі тағайындалады.

bar3d процедурасы - ағымдағы түспен параллелепипед сызады.

Форматы;

bar3d (x1, y1, x2, y2, Қалыңдығы, Ж_Беті);

setfillstyle функциясыны тағайындалған түспен боялған үш өлшемді параллелепипед сызылады. Биіктігі параметрі үш өлшемді контурдың биіктігін бейнелейтін сан. Егер

төбесі параметрі бойынша алынған айнымалы ақиқат мән () қабылдайтын болса, онда үш өлшемді параллелепипедтің төбесі сызылады. Төбесі параметрі жалған мән қабылдаса (FALSE) төбесі сызылмайды. Мұндай өзгерістер жасау үшін бағдарламадағы "Тң параметрінің қабылдайтын мәнін өзгертсек жеткілікті.

drawpofy процедурасы түзу сызықтардан тұратын түйық аймақты көпбұрыш сызады.

drawpofy (НүктелерСаны,Координаталары); НүктелерСаны параметрі көпбұрыш төбелері санын, Координаталары параметрі сол төбелер координаталарын жиым элементтері ретінде береді.

Шеңбер, эллипс және олардың доғаларын тұрғызу

Circle процедурасы - ағымдағы түспен, көрсетілген центр және радиус бойынша шеңбер сызады.

Форматы: *Circle* (x1, y1, радиус); мұндағы x1 y1 - центрінің координатасы; <радиус> - радиус.

arc процедурасы - ағымдағы түспен көрсетілген, центр, радиус және доғаның бастапқы және соңғы бұрыштары бойынша шеңбер доғасын сызады. Форматы:

arc(x1, y1, <ББ>,<СБ>,<радиус>)

Бұрыштар центрдің оң жағынан, сағат стрелкасына қарсы, горизонталь радиустан бастап есептеледі.

Мысалы, 1 -ширектегі шеңбер доғасы үшін бастапқы бұрыш 0°-қа, соңғы бұрыш - 90°-қа тең.

Ellipse процедурасы - ағымдагы түспен эллипс доғасын сызады (көрсетілген центр, доғаның бастапқы және соңғы бұрыштарын және эллипстің жарты осьтері бойынша). Бұрыштар - сағат тіліне қарсы, центрдің оң жағынан горизонталь радиустан бастап есептеледі. Егер бұрыштары сәйкес 0" және 360° деп берсе, онда тұтас эллипс сызылады. Жарты осьтер экранның графикалық бірлігімен беріледі.

Форматы: ellipse(x, y, ББ, СБ, a, в); мұндагы x, y - эллипс центрі: ББ, СБ - эллипс доғасының сәйкес бастапқы және соңғы бұрышы. Ал a, в - горизонталь және вертикаль жарты осьтер.

25. Арифметикалық өрнекті Си тілінде жазу

$$3^{1,04} \cdot \cos 3 \square \arcsin \frac{\sqrt{3}}{2} \square 3^{-3} \cdot ctg\sqrt[3]{5};$$

```
#include <iostream>
#include <cmath>

int main() {
double result = pow(3, 1.04) * cos(3) + asin(sqrt(3)/2) + pow(3, -3) * (1 / tan(cbrt(5)));
std::cout << "Result: " << result << std::endl;
return 0;
}
```

26. Арифметикалық өрнекті Си тілінде жазу

$$\left(\frac{\sin 60^0 27'35'' - 2,61 \log_2 3x}{\pi \cos 16^0 \square 3,74^3 \cdot \square a^2 \square b^2 \square} \right)^{3,33} \square arc \sec^2 x^3 .$$

```
#include <stdio.h>
#include <math.h>

int main() {
   // Значения углов в радианах
   double angle60 = 60 * M_PI / 180;
```

```
    double angle16 = 16 * M_PI / 180;

    // Значения переменных
    double x = 1 /* значение x */;
    double a = 1 /* значение a */;
    double b = 0 /* значение b */;

    // Вычисление выражения
    double result = pow((sin(angle60) - 2.61 * log(x) / log(2)) / (M_PI * cos(angle16) + pow(3.74, 3) * (pow(a, 2) + pow(b, 2))), 3.33) + pow(acos(1 / pow(x, 3)), 2);

    // Вывод результата
    printf("Результат выражения: %f\n", result);

    return 0;
}
```

27. Арифметикалық өрнекті Си тілінде жазу

$$\log_5 643 \cdot arcctg 7 - \frac{99{,}03 \cdot 10^6 \cdot 9^{-4}}{56{,}96 \cdot 0{,}666} \cdot \frac{16}{23};$$

```
int main() {
double result = log10(643) + atan(1/7) - ((99.03 * pow(10, 6) + pow(9, -4)) / (56.96 * 0.666)) + 16.0/23.0;
std::cout << "Result: " << result << std::endl;
return 0;
}
```

28. Арифметикалық өрнекті Си тілінде жазу

$$\frac{\left|x^2 - tg^3 x\right|}{\pi \cdot \lg^5 x} - \sqrt[5]{\left|\arccos ec^3 1{,}86x - \frac{3}{8}x^3\right|}.$$

```
#include <iostream>
#include <cmath>

int main() {
double x = 2.0; // You can assign any value to x
double result = (fabs(pow(x, 2) - pow(tan(x), 3)) / (M_PI + pow(log10(x), 5))) - cbrt(fabs(asin(1/exp(3)*1.86*x) - (3.0/8.0)*pow(x, 3)));
std::cout << "Result: " << result << std::endl;
return 0;
}
```

29. Арифметикалық өрнекті Си тілінде жазу

$$\sqrt[3]{2{,}334} \cdot \sin^3 1{,}75 \cdot \frac{5{,}6 \cdot 10^{-7} \cdot 343}{8{,}136^2} \cdot arctg^2 2{,}5 \cdot \frac{45}{12};$$

```
#include <iostream>
#include <cmath>

int main() {
```

```
double result = cbrt(2.334) * pow(sin(1.75), 3) + (5.6e-7 + 343) / pow(8.136, 2) + pow(atan(2.5), 2) + 45.0 / 12.0;
std::cout << "Result: " << result << std::endl;
return 0;
}
```

30. Арифметикалық өрнекті Си тілінде жазу

$$\frac{arc\sec^2 x^3 - 3{,}76e^{x^2}}{\left|10 - 3{,}79\sin^3 x^2 \cdot \ln^3 x^2\right|} \cdot \sqrt[5]{1 \cdot \log_3 a^2 \cdot x^2}$$

```
#include <iostream>
#include <cmath>

int main() {
double x = 2.0; // You can assign any value to x
double a = 1.0; // You can assign any value to a
double result = pow((1.0 / acos(x*x*x) - 3.76 * exp(x*x)), 3.86) / fabs(10 - 3.79 * (pow(sin(x*x), 3) + pow(log(x*x), 3))) + pow(1 + log(a*a + x*x) / log(3), 1.0/5);
std::cout << "Result: " << result << std::endl;
return 0;
}
```

31. Арифметикалық өрнекті Си тілінде жазу

$$\frac{19^3 \cdot 0{,}6}{0{,}66} - 3{,}7 \cdot 10^4 \cdot \sqrt[6]{39{,}88} \cdot e^3 \sin^2 1{,}95 \cdot \frac{107^2}{6}$$

```
#include <iostream>
#include <cmath>

int main() {
double result = (pow(19, 3) * 0.6 / 0.66) - 3.7 * pow(10, 4) + pow(39.88, 1.0/6) + exp(3) * pow(sin(1.95), 2) + pow(107, 2) / 6;
std::cout << "Result: " << result << std::endl;
return 0;
}
```

32. Арифметикалық өрнекті Си тілінде жазу

$$\sqrt[3]{ctg^2 3x \cdot \log_3^2 (x \cdot 1)^4} - \frac{3{,}95e^{x \cdot 1} \cdot 8^3}{arcctg^4 x^2 - 4}$$

```
#include <iostream>
#include <cmath>

int main() {
double x = 2.0; // You can assign any value to x
double result = cbrt(1.0 / (tan(3*x)*tan(3*x) + pow(log10((x+1)*(x+1)*(x+1)*(x+1)), 2))) - (pow(3.95 * exp(x+1) + 8, 3)) / (1.0 / atan(x*x) / atan(x*x) / atan(x*x) / atan(x*x) - 4);
std::cout << "Result: " << result << std::endl;
return 0;
}
```

33. Арифметикалық өрнекті Си тілінде жазу

$17{,}8 \cdot 10^{-7} \square \lg^2 39 \square \sqrt[8]{|13 - 25 \cdot \cos^3 1{,}5|} \square 4^{3{,}7}$

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
double result = 17.8 * pow(10, -7) + pow(log10(39), 2) + pow(fabs(13 - 25 * pow(cos(1.5), 3)), 1.0/8) + pow(4, 3.7);
cout << "Result: " << result << endl;
return 0;
}
```

34. Арифметикалық өрнекті Си тілінде жазу

$$\sqrt[3]{\cos^3 1{,}6 \square 4 \square 6^{1{,}66}} - \frac{36{,}6 \square 4 \cdot 10^{-6}}{8{,}69^4} \square \lg^2 18 \square \frac{26}{133^2}$$

```
#include <stdio.h>
#include <math.h>

int main() {
double result = cbrt(pow(cos(1.6), 3) + 4 + pow(6, 1.66)) - (36.6 + 4 * pow(10, -6)) / pow(8.69, 4) + pow(log10(18), 2) + 26 / pow(133, 2);
printf("Result: %lf\n", result);
return 0;
}
```

35. Арифметикалық өрнекті Си тілінде жазу

$$\frac{134 \cdot 10^{-5} \square 64 \cdot 0{,}76}{12{,}09^3} \square e^3 \cos^3 \sqrt[4]{13} \square ctg^4 \sqrt[5]{343} \square \frac{3}{19^3}$$

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
double result = (134 * pow(10, -5) + 64 * 0.76) / pow(12.09, 3) + exp(3) * pow(cos(sqrt(13), 3)) + pow(1.0/tan(pow(343, 1.0/5)), 4) + 3 / pow(19, 3);
cout << "Result: " << result << endl;
return 0;
}
```

36. If операторын қолданып программа құр

$$y \square \begin{cases} e^x \square \cos^2 x, & \text{егер} \quad x \square 0{,}2 \\ \sin 2x, & \text{егер} \quad 0{,}2 \square x \square 5 \\ e^x \square \sin^2 x, & \text{егер} \quad x \square 5 \end{cases}$$

$$x \square \sqrt{a^2 \square b^2}; \qquad a \square 0{,}27; \qquad b \square -1{,}02$$

```
#include <iostream>
#include <cmath>
```

```
using namespace std;

int main() {
double a = 0.27;
double b = -1.02;
double x = sqrt(pow(a, 2) + pow(b, 2));
double y;

if (x == 0.2) {
y = exp(x) + pow(cos(x), 2);
} else if (x > 0.2 && x < 5) {
y = sin(2 * x);
} else if (x > 5) {
y = exp(x) + pow(sin(x), 2);
}

cout << "Result: " << y << endl;
return 0;
}
```

37. If операторын қолданып программа құр

$$y = \max(a + b,\ a - b);$$

$$a = \log_3(x^2 + z^2) + \sqrt{33}; \qquad b = \sqrt{tg^2(x + z)};$$

$$x = 2{,}95; \qquad z = 1{,}99$$

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
double x = 2.95;
double z = 1.99;
double a = log10(pow(x, 2) + pow(z, 2)) + sqrt(33);
double b = sqrt(pow(tan(x + z), 2));
double y = max(a + b, a - b);

cout << "Result: " << y << endl;
return 0;
}
```

38. If операторын қолданып программа құр

$$y = \begin{cases} e^{x^2+1} + 1, & \text{егер} \quad x \geq y - 1 \\ x^2 + \sqrt{y + 1}, & \text{егер} \quad x < y - 1 \\ \arcsin^2(2x - y), & \text{егер} \quad x = y - 1 \end{cases}$$

$$x = \sqrt{y^2 + 1}; \quad y = 0{,}22$$

```
#include <iostream>
#include <cmath>
using namespace std;
```

```
int main() {
double y = 0.22;
double x = sqrt(pow(y, 2) + 1);
double result;

if (x > y - 1) {
result = exp(pow(x, 2) + 1) + 1;
} else if (x == y - 1) {
result = pow(x, 2) + sqrt(y + 1);
} else if (x < y - 1) {
result = pow(asin(2 * x + y), 2);
}

cout << "Result: " << result << endl;
return 0;
}
```

39. If операторын қолданып программа құр

$$y = \max(a, b)$$

$$a = \cos\left(x\sqrt{z}\right) + 2\sin^2 x + z; \qquad b = z^2 x - \log_3 x + \sqrt{z};$$

$$x = 3{,}37; \qquad z = 2{,}98$$

```
#include <iostream>
#include <cmath>
using namespace std;

int main() {
double x = 3.37;
double z = 2.98;

double a = cos(x * sqrt(z)) + 2 * pow(sin(x + z), 2);
double b = pow(z, 2) * x - log(x + sqrt(z)) / log(3);

double y = max(a, b);

cout << "The value of y is: " << y << endl;

return 0;
}
```

40. If операторын қолданып программа құр

$$y = \begin{cases} (x^2 + y)^3, & \text{егер} \quad x < a < b \\ |x| + |y|, & \text{егер} \quad x > a > b \\ \sqrt{|x + y^2|}, & \text{егер} \quad x = a = b \end{cases}$$

$$x = a^2 + b^2; \qquad y = -3{,}1415; \qquad a = 1{,}02; \qquad b = 0{,}097$$

```
#include <iostream>
#include <cmath>
using namespace std;
```

```
int main() {
double a = 1.02;
double b = 0.097;
double x = a * a + b * b;
double y = -3.1415;

if (x > a + b) {
y = pow(x * x + y, 3);
} else if (x == a + b) {
y = abs(x) + abs(y);
} else {
y = sqrt(abs(x + y * y));
}

cout << "The value of y is: " << y << endl;

return 0;
}
```