**PROGRAM 1:**

#include <iostream>

using namespace std;

int main()

{

int a = 100;//get the value,

cout<< a; //prints 100

//get the memory address

cout<< &a; //prints 1024

}

**OUTPUT:**

1000x6dfefc

**PROGRAM 2:**

#include <iostream>

using namespace std;

int main()

{

int a, b;

a = 88;

b = 100;

cout << "The address of a is: " << &a << endl;

cout << "The address of b is: " << &b << endl;

return 0;

}

**Output:**

The address of a is: 0x6dfefc

The address of b is: 0x6dfef8

**Program 3:**

#include <iostream>

using namespace std;

int main()

{

int a = 100;

int \*p = &a;

cout << a << " " << &a <<endl;

cout << p << " " << &p <<endl

return 0;

}

**Output:**

100 0x6dfefc

0x6dfefc 0x6dfef8

**Program 4:**

#include <iostream>

using namespace std;

int main()

{

int a = 100;

int \*p = &a;

cout << a << endl;

cout << &a << endl;

cout << p << " " << \*p << endl;

cout << &p << endl;

return 0;

}

**Output:**

100

0x6dfefc

0x6dfefc 100

0x6dfef8

**Program 5:**

#include <iostream>

using namespace std;

int main()

{

int a = 100, b = 88, c = 8;

int \*p1 = &a, \*p2, \*p3 = &c;

p2 = &b; // p2 points to b

p2 = p1; // p2 points to a

b = \*p3; //assign c to b

\*p2 = \*p3; //assign c to a

cout << a << b << c;

return 0;

}

**Output:**

888

**PROGRAM 6:**

#include <iostream>

using namespace std;

int main ()

{

int value1 = 5, value2 = 15;

int \*p1, \*p2;

p1 = &value1; // p1 = address of value1

p2 = &value2; // p2 = address of value2

\*p1 = 10; // value pointed to by p1=10

\*p2 = \*p1; // value pointed to by p2= value // pointed to by p1

p1 = p2; // p1 = p2 (pointer value copied)

\*p1 = 20; // value pointed to by p1 = 20

cout << "value1==" << value1 << "/ value2==" << value2;

return 0;

}

**OUTPUT:**

value1==10/ value2==20

**PROGRAM 7:**

#include <iostream>

using namespace std;

int main ()

{

int a=3;

int \*pa=&a;// sizeof returns the # of bytes…

cout<<"\n pa= "<<pa<< " sizeof(pa): "<<sizeof(pa);

cout<<"\n \*pa= "<<\*pa<<" sizeof(\*pa): "<<sizeof(\*pa);

cout<<"\n &pa= "<<&pa<<" sizeof(&pa): "<<sizeof(&pa);

return 0;

}

**Output:**

pa= 0x6dfefc sizeof(pa): 4

\*pa= 3 sizeof(\*pa): 4

&pa= 0x6dfef8 sizeof(&pa): 4

**PROGRAM 8:**

#include <iostream>

using namespace std;

int main ()

{

char c='z';

char \*pc=&c;

// sizeof returns the # of bytes…

cout<<"\n pc= "<<pc<< " sizeof(pc): "<<sizeof(pc);

cout<<"\n \*pc= "<<\*pc<<" sizeof(\*pc): "<<sizeof(\*pc);

cout<<"\n &pc= "<<&pc<<" sizeof(&pc): "<<sizeof(&pc);

return 0;

}

**Output:**

pc= z  m sizeof(pc): 4

\*pc= z sizeof(\*pc): 1

&pc= 0x6dfef8 sizeof(&pc): 4

**Program 9:**

#include <iostream>

using namespace std;

int main ()

{

double d=1.03;

double \*pd=&d;

// sizeof returns the # of bytes…

cout<<"\n pd= "<<pd<< " sizeof(pd): "<<sizeof(pd);

cout<<"\n \*pd= "<<\*pd<<" sizeof(\*pd): "<<sizeof(\*pd);

cout<<"\n &pd= "<<&pd<<" sizeof(&pd): "<<sizeof(&pd);

return 0;

}

**Output:**

pd= 0x6dfef8 sizeof(pd): 4

\*pd= 1.03 sizeof(\*pd): 8

&pd= 0x6dfef4 sizeof(&pd): 4

**PROGRAM 10:**

#include <iostream>

using namespace std;

int main ()

{

int a=3;

char c='z';

double d=1.03;

int \*pa=&a;

char \*pc=&c;

double \*pd=&d; // sizeof returns the # of bytes…

cout<<" \nsizeof(pa): "<<sizeof(pa); cout<<" sizeof(\*pa): "<<sizeof(\*pa);

cout<<" sizeof(&pa): "<<sizeof(&pa);

cout<<" \nsizeof(pc): "<<sizeof(pc); cout<<" sizeof(\*pc): "<<sizeof(\*pc);

cout<<" sizeof(&pc): "<<sizeof(&pc);

cout<<" \nsizeof(pd): "<<sizeof(pd); cout<<" sizeof(\*pd): "<<sizeof(\*pd);

cout<<" sizeof(&pd): "<<sizeof(&pd);

return 0;

}

**Output:**

sizeof(pa): 4 sizeof(\*pa): 4 sizeof(&pa): 4

sizeof(pc): 4 sizeof(\*pc): 1 sizeof(&pc): 4

sizeof(pd): 4 sizeof(\*pd): 8 sizeof(&pd): 4

**Program 11:**

#include <iostream>

using namespace std;

int main ()

{

int m = 10;

int &j = m; // j is a reference variable

cout <<"value of m = "<< m << endl;

//print 10

j = 18;

cout << "value of m ="<< m << endl;

// print 18

return 0;

}

**Output:**

value of m = 10

value of m =18

**Program 12:**

#include <iostream>

using namespace std;

void IndirectSwap(char \*Ptr1, char \*Ptr2){

char temp = \*Ptr1;

\*Ptr1 = \*Ptr2;

\*Ptr2 = temp;

}

int main() {

char a = 'y';

char b = 'n';

IndirectSwap(&a, &b);

cout << a << b << endl;

return 0;

}

**Output:**

Ny

**Program 13:**

#include <iostream>

using namespace std;

void IndirectSwap(char& y, char& z) {

char temp = y;

y = z;

z = temp;

}

int main() {

char a = 'y';

char b = 'n';

IndirectSwap(a, b);

cout << a << b << endl;

return 0;

}

**Output:**

Ny

Program 14:

#include <iostream>

using namespace std;

int main ()

{

int a[5];

cout << "Address of a[0]: " << &a[0] << endl

<< "Name as pointer: " << a << endl;

return 0;

}

**Output:**

Address of a[0]: 0x6dfeec

Name as pointer: 0x6dfeec

**Program 15:**

#include <iostream>

using namespace std;

int main()

{

int a[5] = {2,4,6,8,22};

cout << \*a << " "

<< a[0];

return 0;

} //main

**Output:**

2 2

**Program 16:**

#include <iostream>

using namespace std;

int main(){

int a[5] = {2,4,6,8,22};

int \*p = a;

cout << a[0] << " "

<< \*p;

return 0;

}

Output:

2 2

**Program 17:**

#include <iostream>

using namespace std;

int main()

{

int a[5] = {2,4,6,8,22};

int \*p = &a[1];

cout << a[0] << " "

<< p[-1];

cout << a[1] << " "

<< p[0];

return 0;

}

**Output:**

2 24 4

**Program 18:**

#include <iostream>

using namespace std;

int main()

{

int list[5] = {9, 8, 7, 6, 5};

int \*p;

p = list;//points to 1st entry

cout<<p<<"\n";

p = &list[0];//points to 1st entry

cout<<p<<"\n";

p = &list[1];//points to 2nd entry

cout<<p<<"\n";

p = list + 1; //points to 2nd entry

cout<<p;

return 0;

}

**Output:**

0x6dfee8

0x6dfee8

0x6dfeec

0x6dfeec

**Program 19:**

#include <iostream>

using namespace std;

int main()

{

int \*p;

p = 0;

cout << p << endl; //prints 0

cout << &p << endl;//prints address of p

cout << \*p << endl;//Error!

return 0;

}

**Output:**

0

0x6dfefc

Program 20:

#include <iostream>

using namespace std;

void initialize(int list[], int size, int value)

{

for(int i=0; i<size; i++)

list[i] = value;

}

void print(int list[], int size)

{

cout << "[ ";

for(int i=0; i<size; i++)

cout << list[i] << " ";

cout << "]" << endl;

}

int\* addElement(int list[], int& size, int value)

{

int\* newList = new int [size+1]; // make new array

if(newList==0)

{

cout << "Memory allocation error for addElement!" << endl;

return 0;

}

for(int i=0; i<size; i++)

newList[i] = list[i];

if(size) delete [] list;

newList[size] = value;

size++;

return newList;

}

// for deleting the first element of the array

int\* deleteFirst(int list[], int& size)

{

if(size <= 1)

{

if( size) delete list;

size = 0;

return NULL;

}

int\* newList = new int [size-1]; // make new array

if(newList==0)

{

cout << "Memory allocation error for deleteFirst!" << endl;

return 0;

}

for(int i=0; i<size-1; i++) // copy and delete old array

newList[i] = list[i+1];

delete [] list;

size--;

return newList;

}

int main()

{

cout << "Enter list size: ";

int n;

cin >> n;

int \*A = new int[n];

if(n<=0)

{

cout << "bad size" << endl;

return 0;

}

initialize(A, n, 0); // initialize the array A with value 0

print(A, n);

A = addElement(A,n,5); //add an element of value 5 at the end of A

print(A, n);

A = deleteFirst(A,n); // delete the first element from A

print(A, n);

delete [] A;

return 0;

}

Output:

Enter list size: 5

[ 0 0 0 0 0 ]

[ 0 0 0 0 0 5 ]

[ 0 0 0 0 5 ]

Program 21:

#include <iostream>

using namespace std;

void initialize(int list[], int size, int value)

{

for(int i=0; i<size; i++)

list[i] = value;

}

void print(int list[], int size)

{

cout << "[ ";

for(int i=0; i<size; i++)

cout << list[i] << " ";

cout << "]" << endl;

}

// for adding a new element to end of array

// here “list” is a reference to a pointer variable: if the value of the pointer is changed in function, the change is global.

void addElement( int \* & list, int & size, const int value ){

int \* newList = new int [size + 1];

if( newList == NULL ){

cout << "Memory allocation error for addElement!" << endl;

return;

}

for( int i = 0; i < size; i++ )

newList[ i ] = list[ i ];

if( size ) delete [] list;

newList[ size ] = value;

size++;

list = newList;

return;

}

// for deleting the first element of the array

void deleteFirst( int \* & list, int & size )

{

if( size <= 1 )

{

if( size )

delete list;

list = NULL;

size = 0;

return;

}

delete list; // delete the first element

list++;

size--;

return;

}

int main()

{

int \* A = NULL;

int size = 0;

int i;

for( i = 0; i < 10; i++ )

addElement( A, size, i );

for( i = 0; i < 10; i++ )

cout << A[i] << " ";

cout << endl;

for( i = 0; i < 4; i++ )

deleteFirst( A, size );

for( i = 0; i < 6; i++ )

cout << A[i] << " ";

cout << endl;

return 0;

}

Output:

0 1 2 3 4 5 6 7 8 9

4 5 6 7 8 9

Program 22:

#include<iostream>

using namespace std;

int main()

{ int \*\*table;

table = new int\*[6];

table[0]= new int[3];

table[1]= new int[1];

table[2]= new int[5];

table[3]= new int[10];

table[4]= new int[2];

table[5]= new int[6];

table[0][0] = 1; table[0][1] = 2; table[0][2] = 3;

table[1][0] = 4;

table[2][0] = 5; table[2][1] = 6; table[2][2] = 7; table[2][3] = 8; table[2][4] = 9;

table[4][0] = 10; table[4][1] = 11;

cout << table[2][5] << endl;

return 0;

}

Output:

0

Program 23:

#include<iostream>

using namespace std;

int main()

{

int table[3][4] = {{1,2,3,4},{5,6,7,8},{9,10,11,12}};

for(int i=0; i<3; i++)

{

for(int j=0; j<4; j++)

cout << \*(\*(table+i)+j)<<"\t";;

cout << endl;

}

return 0;

}

Output:

1 2 3 4

5 6 7 8

9 10 11 12