* Pointers are variables which store the address of other variables.
* &- address of

Int main()

Int i= 10;

Int \*p=&i; //stores address of i

Cout<<p; //prints address

Cout<<\*p; //prints 10

int a = 50;

int \*ptr = &a;

int \*q = ptr;

(\*q)++;

cout << a << endl;

op:51

int a = 50;

int \*ptr = &a;

cout << (\*ptr)++ << “ “;

cout << a << endl;

50 51

int \*ptr = 0;

int a = 10;

\*ptr = a;

cout << \*ptr << endl;

error

float f = 10.5;

float p = 2.5;

float\* ptr = &f;

(\*ptr)++;

\*ptr = p;

cout << \*ptr << “ “ << f << “ “ << p;

2.5 all

int a = 7;

int \*c = &a;

c = c + 1;

cout << a << " " << \*c << endl;

7 garbage

Assume memory address of variable ‘a’ is : 400 (and an integer takes 4 bytes), what will be the output -

int a = 7;

int \*c = &a;

c = c + 3;

cout << c << endl;

412

int a[] = {1, 2, 3, 4};

cout << \*(a) << " " << \*(a+1);

1 2

int a[6] = {1, 2, 3};

cout << a << “ “ << &a;

200 200

int a[6] = {1, 2, 3};

cout << (a + 2);

208

int a[6] = {1, 2, 3};

int \*b = a;

cout << b[2];

3

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

cout << \*(a) << " " << \*(a + 4);

1 5

int a[] = {1, 2, 3, 4};

int \*p = a++;

cout << \*p << endl;

error

char ch = 'a';

char\* ptr = &ch;

ch++;

cout << \*ptr << endl;

b

#### Assume address of 0th index of array ‘b’ is 200. What is the output -

char b[] = "xyz";

char \*c = &b[0];

cout << c << endl;

xyz

char b[] = "xyz";

char \*c = &b[0];

c++;

cout << c << endl;

yz

char s[]= "hello";

char \*p = s;

cout << s[0] << " " << p[0];

h h

void changeSign(int \*p){

\*p = (\*p) \* -1;

}

int main(){

int a = 10;

changeSign(&a);

cout << a << endl;

}

-10

void fun(int a[]) {

cout << a[0] << " ";

}

int main() {

int a[] = {1, 2, 3, 4};

fun(a + 1);

cout << a[0];

}

2 1

void square(int \*p){

int a = 10;

p = &a;

\*p = (\*p) \* (\*p);

}

int main(){

int a = 10;

square(&a);

cout << a << endl;

}

10

int a = 10;

int \*p = &a;

int \*\*q = &p;

int b = 20;

\*q = &b;

(\*p)++;

cout << a << " " << b << endl;

20 21

int a = 100;

int \*p = &a;

int \*\*q = &p;

int b = (\*\*q)++ + 4;

cout << a << " " << b << endl;

101 104

int a = 100;

int \*p = &a;

int \*\*q = &p;

int b = (\*\*q)++;

int \*r = \*q;

(\*r)++;

cout << a << " " << b << endl;

102 100

void increment(int \*\*p){

(\*\*p)++;

}

int main(){

int num = 10;

int \*ptr = &num;

increment(&ptr);

cout << num << endl;

}

11

#include <iostream>

using namespace std;

int main()

{

int arr[] = {4, 5, 6, 7};

int \*p = (arr + 1);

cout << \*arr + 9;

return 0;

}

13

int main ()

{

int numbers[5];

int \* p;

p = numbers;

\*p = 10;

p = &numbers[2];

\*p = 20;

p--;

\*p = 30;

p = numbers + 3;

\*p = 40;

p = numbers;

\*(p+4) = 50;

for (int n=0; n<5; n++) {

cout << numbers[n] << ",";

}

return 0;

}

10 30 20 40 50

int main()

{

char \*ptr;

char Str[] = "abcdefg";

ptr = Str;

ptr += 5;

cout << ptr;

return 0;

}

Fg

int main()

{

char arr[20];

int i;

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

\*(arr + i) = 65 + i;

}

\*(arr + i) = '\0';

cout << arr;

return 0;

}

ABCDEFGHIJ

void swap (char \*x, char \*y)

{

char \*t = x;

x = y;

y = t;

}

int main()

{

char \*x = "geeksquiz";

char \*y = "geeksforgeeks";

char \*t;

swap(x, y);

cout<<x << " "<<y;

t = x;

x = y;

y = t;

cout<<" "<<x<< " "<<y;

return 0;

}

Gq gfg gfg gq

int main()

{

float arr[5] = {12.5, 10.0, 13.5, 90.5, 0.5};

float \*ptr1 = &arr[0];

float \*ptr2 = ptr1 + 3;

cout<<\*ptr2<<" ";

cout<< ptr2 - ptr1;

return 0;

}

90.5 3

int main() {

char st[] = "ABCD";

for(int i = 0; st[i] != ‘\0’; i++) {

cout << st[i] << \*(st)+i << \*(i+st) << i[st];

}

return 0;

}

A65AAB66BBC67CCD68DD

void Q(int z)

{

z += z;

cout<<z << " ";

}

void P(int \*y)

{

int x = \*y + 2;

Q(x);

\*y = x - 1;

cout<<x << " ";

}

int main()

{

int x = 5;

P(&x);

cout<<x;

return 0;

}

14 7 6

int main()

{

int \*\*\*r, \*\*q, \*p, i=8;

p = &i;

(\*p)++;

q = &p;

(\*\*q)++;

r = &q;

cout<<\*p << " " <<\*\*q << " "<<\*\*\*r;

return 0;

}

10 10 10

int f(int x, int \*py, int \*\*ppz) {

int y, z;

\*\*ppz += 1;

z = \*\*ppz;

\*py += 2;

y = \*py;

x += 3;

return x + y + z;

}

int main() {

int c, \*b, \*\*a;

c = 4;

b = &c;

a = &b;

cout << f(c, b, a);

return 0;

}

19

#### Void pointer can point to which type of objects ? int float double all

#### What is the correct syntax of declaring and defining a reference? Int a=10; int &b= a;

#### Create array: int \*arr=new int[5]; delete: delete[] arr

#### On deleting a dynamic memory, if the pointer value is not modified, then the pointer points to? Same deleted memory location

#### Int \*\*twodarray=new int\*[10];

* for(int i = 0; i < 10; i++)
* delete [] twoDArray[i];

#### Inline functions are used to reduce the function call overhead. Inline function is a function that is expanded in line when it is called. Used when function is small

#### All parameters of a function can be default parameters

int main()

{

int d = 65;

int i = d;

char ch = i;

cout << ch << endl;

}

A

void updateValue(int \*p){

\*p = 610 % 255;

}

int main(){

char ch = 'A';

updateValue((int\*)&ch);

cout << ch;

}

D

void func(int p){

cout << p << " ";

}

int main(){

int i = 10;

int &p = i;

func(p++);

cout << i;

}

10 11

void func(int i, int& j, int p){

i++;

j++;

p++;

}

int main(){

int i = 10;

int j = 6;

int &p = i;

func(i, j, p);

cout << i << " " << j << " " << p;

}

10 7 10

int x = 1;

void print(){

int x = 2;

{

int x = 3;

cout << x << endl;

}

}

int main(){

print();

return 0;

}

3

#define MULTIPLY(a, b) a\*b

int main(){

cout << MULTIPLY(2+3, 3+5);

return 0;

}

16

#define SQUARE(x) x\*x

int main(){

int x = 36 / SQUARE(6);

cout << x;

return 0;

}

36

int getValue(int x = 0, int y = 0, int z){

return (x + y + z);

}

int main(){

cout << getValue(10);

return 0;

}

Default parameters must be added last in C++ functions, compilation error

int main(){

int const p = 5;

cout << ++p;

return 0;

}

Error

int main(){

int p = 5;

int const \*q = &p;

p++;

cout << p << endl;

return 0;

}

6 Value of p can be increased or decreased as it is not a const value.

int main(){

int p = 5;

int \* const q = &p;

}

Q++ will give error

int main(){

int p = 5;

int const \*q = &p;

}

(\*q)++ error

int main(){

int p = 5;

int const \* const q = &p;

}

Q++, (\*q)++ will give error

int main(){

const int p = 5;

int const \*q = &p;

}

P++, (\*q)++