LAB MANUAL 10-A

POINTERS

WHAT ARE POINTERS?

A **pointer** is a variable whose value is the address of another variable. Like any variable or constant, you must declare a pointer before you can work with it. The general form of a pointer variable declaration is −

```
type *var-name;
```

Here, **type** is the pointer's base type; it must be a valid C++ type and **var-name** is the name of the pointer variable. The asterisk you used to declare a pointer is the same asterisk that you use for multiplication. However, in this statement the asterisk is being used to designate a variable as a pointer. Following are the valid pointer declaration −

```
int    *ip;    // pointer to an integer
double *dp;    // pointer to a double
float  *fp;    // pointer to a float
char   *ch     // pointer to character
```

The actual data type of the value of all pointers, whether integer, float, character, or otherwise, is the same, a long hexadecimal number that represents a memory address. The only difference between pointers of different data types is the data type of the variable or constant that the pointer points to.

USING POINTERS IN C++

There are few important operations, which we will do with the pointers very frequently. **(a)** We define a pointer variable. **(b)** Assign the address of a variable to a pointer. **(c)** Finally access the value at the address available in the pointer variable. This is done by using unary operator * that returns the value of the variable located at the address specified by its operand. Following example makes use of these operations −

Example

```
#include <iostream>

using namespace std;

int main () {
   int  var = 20;    // actual variable declaration.
   int  *ip;         // pointer variable

   ip = &var;        // store address of var in pointer variable

   cout << "Value of var variable: ";
   cout << var << endl;

   // print the address stored in ip pointer variable
   cout << "Address stored in ip variable: ";
   cout << ip << endl;

   // access the value at the address available in pointer
   cout << "Value of *ip variable: ";
   cout << *ip << endl;

   return 0;
}
```

When the above code is compiled and executed, it produces result something as follows −

```
Value of var variable: 20
Address stored in ip variable: 0xbfc601ac
Value of *ip variable: 20
```

Question 1

What will be the output of the following program?

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

int main()
{
    int a = 32, *ptr = &a;
    char ch = 'A', &cho = ch;

    cho += a;
    *ptr += ch;
    cout << a << ", " << ch << endl;
    return 0;
}
```

Solution

Answer: c. 129, a

Explanation: The “ptr” variable is a pointer which holds the address of variable “a”. And “*ptr” returns the value of “a” variable. “cho” is a reference variable to “ch”. So any change made to “cho” will be reflected to “ch”. As such, when “cho” is increased by 32, it adds to the ASCII value of “A”(which is 65), and this results to 97 which is the ASCII value of “a”(from the alphabet). So this “a” gets stored in “ch”. As for when “*ptr” is incremented by “ch”, it gives value 97+32=129.

Question 2

Write a C++ program to reverse an array using pointers.