ISSN: 2249-7137 Vol. 10 Issue 5, May 2020 Impact Factor: SJIF 2020 = 7.13

ACADEMICIA
An International Multidisciplinary Research Journal

ISSN (online) : 2249 –7137

Editor-in-Chief : Dr. B.S. Rai

Impact Factor : SJIF 2020 = 7.13

Frequency : Monthly

Country : India

Language : English

Start Year : 2011

Indexed/ Abstracted : Scientific Journal Impact Factor (SJIF2020 - 7.13), Google Scholar, CNKI Scholar, EBSCO Discovery, Summon (ProQuest), Primo and Primo Central, I2OR, ESJI, IIJIF, DRJI, Indian Science and ISRA-JIF and Global Impact Factor 2019 - 0.682

E-mail id: saarjjournal@gmail.com

VISION

The vision of the journals is to provide an academic platform to scholars all over the world to publish their novel, original, empirical and high quality research work. It propose to encourage research relating to latest trends and practices in international business, finance, banking, service marketing, human resource management, corporate governance, social responsibility and emerging paradigms in allied areas of management including social sciences , education and information & technology. It intends to reach the researcher's with plethora of knowledge to generate a pool of research content and propose problem solving models to address the current and emerging issues at the national and international level. Further, it aims to share and disseminate the empirical research findings with academia, industry, policy makers, and consultants with an approach to incorporate the research recommendations for the benefit of one and all.

ISSN: 2249-7137 Vol. 10 Issue 5, May 2020 Impact Factor: SJIF 2020 = 7.13

ACADEMICIA
An International Multidisciplinary Research Journal
(Double Blind Refereed & Reviewed International Journal)

ACADEMICIA

ISSN: 2249-7137 Vol. 10, Issue 5, May 2020 Impact Factor: SJIF 2020 = 7.13

ACADEMICIA
An International Multidisciplinary Research Journal
(Double Blind Refereed & Reviewed International Journal)

DOI: 10.5958/2249-7137.2020.00270.0

USING THE CAPABILITIES OF MODERN PROGRAMMING LANGUAGES IN SOLVING PROBLEMS OF TECHNICAL SPECIALTIES

Ergashev Nuriddin Gayratovich*; Shukurov Akmal Uktamovich; Jabborov ElbekErkino'g'li*****

*Karshi Engineering, Economics Institute, UZBEKISTAN
Email id: finaledition2@mail.ru

**Karshi Engineering - Economics Institute, UZBEKISTAN
Email id: specialist0202@mail.ru

***Karshi Engineering - Economics Institute, UZBEKISTAN
Email id: finaledition2@mail.ru

ABSTRACT

This article provides recommendations and examples of how to use modern capabilities of programming languages in technical specialization, as well as to solve these problems on the example of the C ++ programming language. In addition, the C ++ programming language describes the array, its types, capabilities, array problems, and several programming solutions.

KEYWORDS: *C ++, Low-Level Programming Language, High-Level Programming Language, Operators, Arrays, Multidimensional Arrays, Static Arrays, Dynamic Arrays, Character Arrays, Word Arrays, Functions, Alloc, Malloc, Size Of, Null, New .*

INTRODUCTION

Accelerating the updating of modern educational content in the field of programming, the formation of teaching methods, programming tools and methodologies is facing challenges. The development of electronic devices, the systems that support them, and the increasing complexity of software and programming languages pose new challenges to the education system. Constant improvement of computer programs forces the user to change the existing approaches to their development.

Modern computers were initially able to perform the simplest arithmetic and logic operations. In order for a machine to solve a complex problem, it must have a program in its memory. As you know, a program is a specific sequence of machine codes that controls the operation of a specific computing tool. Hundreds of programming languages have been created to make the software creation process easier.

The goal. In addition to specialization programs, specialization programs in technical universities can be completed using the capabilities of programming languages. This requires a good knowledge of programming language skills. Teaching students to specialize in technical programming by using modern programming languages and using the capabilities of programming languages. In this way, students learn the basics of both specialization and programming languages. Nowadays, programming languages are the basis of the management of any electronic devices[1].

Scientific novelty of the article. Operators used in a program based on algorithms must meet the requirements of structured programming. It has no input / output, no dynamic memory allocation, no parallel computing. These tools are implemented through the external functions of the program. The following is an example of a C ++ programming language that analyzes the structure, types, and several examples of programming languages.

C ++ programming serves to increase the performance of computers and operating system capabilities. That's why it's designed for professional programmers. The C ++ programming language is related to the Unix operating system, and many of the programs used in this system were originally written in C ++. The C ++ programming language allows you to take full advantage of all the features of a computer. In C ++, programs represent actions that result in data being retrieved. In the program, actions are given by operators. Data is made by identifying and describing objects. It is necessary to describe the objects used in the program. The description links the object and some of its characteristics. These characteristics include type, designation, memory class, range of motion, and initial values.

The main types in the programming language are integers and floating point numbers. It also creates full versions of data from pointers, arrays, and operands. An expression consists of operands and pointers. Each expression can be an instruction, including a call to a value transfer function. Indicators determine the arithmetic of a machine with an unrelated address. In C ++, the instructor ({}), conditional branching (if), selecting a long-standing alternative (switch), upward repetition (for, while), downward repetition (do), as well as repetition There are control structures such as break. The disadvantage of this language is the inconvenience of the syntax of these language structures.

In C ++, arrays can be used to perform some complex types of tasks, and a number of conveniences are created when performing these tasks. An array is an ordered set of finite values of the same category. Examples of arrays are vectors and matrices from a math course. Arrays are generally divided into one-dimensional, multidimensional, dynamic, and static types. An array is said to be one-dimensional if its element can be referenced by an index. The index of array elements in C ++ programming languages always starts from zero.

Multidimensional static arrays. There are no restrictions on the type of array element in C ++, but these types must be of a finite size. Because the compiler must be able to calculate how much free space (bytes) the array takes up in memory. In particular, the component of an array can be

an array, resulting in a two-dimensional array called a matrix. If the elements of the matrix are also vectors, the result is a three-dimensional array. This way you can create arrays of any size, depending on the task at hand. In a two-dimensional array, the first index indicates the number of rows, and the second the number of columns[2].

Results and practical applications. Declaring two-dimensional static arrays is as follows: type array_name [array_strings_number] [array_columns_number];

When declaring two-dimensional static arrays, the difference from one-dimensional is that the array name is followed by two values in parentheses ([]). The first of these represents the number of rows, and the second the number of columns. That is, a two-dimensional array element is referenced by two indices. Two-dimensional arrays are reminiscent of matrices known from the mathematics course. Example of declaring a two-dimensional array:

int a[3][3], b[2][4];

A matrix B matrix

a00 a01 a02 b00 b01 b02 b03

a10 a11 a12 b10 b11 b12 b13

a20 a21 a22

A matrix has 3 rows and 3 columns. The matrix B has 2 rows and 4 columns. In two-dimensional arrays, index 1 represents the row, and index 2 represents the column. The first element of the first line a10 - a is read as one zero element. a is not called ten. An array (mxn) with m rows and n columns is called a dimensional array. If m = n (the number of rows and columns are equal), the square is called an array. The following are some examples of initializing multidimensional arrays:

int a[2][2]={1,5,7,2};

int b[2][3]={ {0,2,3}, {4,5,6} };

For example. A program that outputs the matrix to the sum of rows and arrays.

```
#include<iostream>
using namespace std;
int main()
{
long n,m,a[100][100],w[100],s[100];
cout<<"enter the size of the matrix columns =";cin>>n;
cout<<"enter the size of the matrix rows =";cin>>m;
for (int i=1;i<=n;i++)
 {for (int j=1;j<=m;j++)
{cin>>a[i][j];} }
```

```
for (int j=1;j<=m;j++) s[j]=0;
for (int j=1;j<=m;j++)
{for (int i=1;i<=n;i++)
{
s[j]+=a[i][j];
 }
 }
for (int j=1;j<=m;j++) {
        a[n+1][j]=s[j];
}
for (int i=1;i<=n;i++) w[i]=0;
for (int i=1;i<=n;i++)
{
for (int j=1;j<=m;j++)
{
w[i]+=a[i][j];
}
}
for (int i=1;i<=n;i++)
{
a[i][m+1]=w[i];
}
for (int i=1;i<=n;i++)
{
for (int j=1;j<=m;j++)
{
cout<<a[i][j]<<" ";
} cout<<a[i][m+1];
cout<<endl;}
for (int j=1;j<m;j++) cout<<a[n+1][j]<<" ";
cout<<a[n+1][m];
return 0;}
```

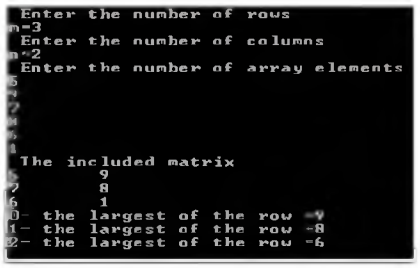
```
 enter the size of the matrix columns =3
 enter the size of the matrix rows =2
6
5
1
7
8
4
6 5 11
4 7 11
8 9 17
18 21
```

The result:

You can send an array as a function parameter and get an array as a function result. When passing a matrix to a function, you must also send the number of rows and columns of the matrix by its name. There are several ways to use an array in a function, some of which we will look at. Transfer the matrix to the function:

```
#include <iostream>
#include<math.h>
using namespace std;
void matrix_print(int a[10][10], int m, int n){
//output the matrix in tabular form
for (int i = 0; i < m; i++){
for (int j = 0; j < n; j++){
cout << a[i][j] << "\t";}
cout << "\n";}}
int row_max(int a[], int n){
//determine the largest element of the array
int max = a[0];
for (int i = 1; i < n; i++)
if (max < a[i]) max = a[i];
return max;}
int main(){
int  m, n, a[10][10];
cout << "Enter the number of rows \nm=";  cin >> m;
cout << "Enter the number of columns \nn="; cin >> n;
cout <<"Enter the number of array elements \n";
for (int i = 0; i < m; i++)
for (int j = 0; j < n; j++)
cin >> a[i][j];
cout << "The included matrix \n";
//the number of matrices, rows, and columns is sent to the function
matrix_print(a, m, n);
for (int i = 0; i < m; i++){
// the address of the 0th element of the i-th string to the function
```

```
// and we send the number of elements
cout << i << "-the largest of the row =" <<
row_max(&a[i][0], n);
cout << endl;}
return 0;
system ("pause");}
```



The result:

Dynamic arrays. The disadvantage of static arrays is that their elements need to be declared in advance. In addition, these elements are limited by the size of the memory segment allocated to the data. Second, a large enough array may be declared and the allocated memory may not be fully utilized to solve the problem. These shortcomings can be overcome by using dynamic arrays. Because they allow you to create arrays of the size you need during program execution and delete them when you don't need them. You can use the malloc (), calloc () functions, or the new operator to allocate memory to dynamic arrays. The delete operator is used to free up memory allocated to a dynamic object[3].

The above functions are located in the <<alloc.h>> library. The syntax of the Malloc () function is Void * malloc (size_t size), which separates a continuous byte-sized area from memory. If memory allocation is successful, the malloc () function returns the start address of the allocated field. If the required memory allocation is successful, the function returns NULL. As you can see from the syntax, the function returns a value of type void. In practice, it is necessary to allocate memory for the object. This is done using a specific type of void conversion technology. For example, allocating space to an array of length 3 for an entire type can be done as follows:

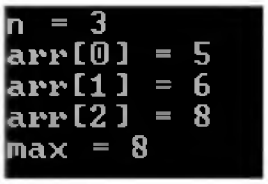
int * pint=(int*)malloc(3*sizeof(int));

The calloc () function, unlike the malloc function, initializes array elements with a value of 0 in addition to allocating space for the array.

The syntax of this function is: Void * calloc (size_tnum, size_t size); The num parameter tells you how many elements are in the allotted area and the size of each element. The Free () memory release function has a single parameter that points to the partition to be erased. Void free (void * block); Having the Free () function set to void allows you to delete any type of memory. The following is a program for finding the largest of the elements of an array for dynamic arrays:

```
#include <iostream>
using namespace std;
intmain()
{
int *arr;
int size;
```

```
cout<< "n = ";
cin>> size;
if (size <= 0) {
cerr<< "Invalid size" <<endl;
return 1;
}
arr = new int[size];
for (inti = 0; i< size; i++) {
cout<< "arr[" <<i<< "] = ";
cin>>arr[i];
}
int max = arr[0];
for (inti = 1; i< size; i++) {
if (arr[i] > max) {
max = arr[i];
}
}
cout<< "max = " << max <<endl;
delete [] arr; // free up memory
return 0;
}
```



The result:

The C ++ programming language also uses other types of arrays, which are analyzed below with examples. Array pointers are defined as: <type> * <name> [<number>]. For example, the definition int * pt [9] inserts an array of nine elements into objects of type int. An array of pointers is useful for describing arrays of rows. For example, a two-dimensional array is used to enter a list of surnames:

Char fam[][20]= {“ Ergashev”,”Shukurov”,”Jabborov”}

The memory consists of 60 elements, as each last name is filled with 0's. Using an array of pointers, this array can be described as follows.

Char *pf[]= {“ Ergashev”,”Shukurov”,”Jabborov”}

The list here contains 23 items in memory, as each last name is marked with a 0 at the end. Arrays of pointers allow you to sort complex elements in a simple way. In the following example, the first elements of the matrix rows are displayed in ascending order. Here, an array of

auxiliary pointers is created, the array is sorted, and matrix elements are output based on the array.

IN CONCLUSION

The purpose of the above considerations and considerations is not to learn a programming language, but to develop algorithmic thinking. Introducing a variety of ways and means of thinking that can be used to solve some problems. Another of these approaches is to get acquainted with these programming languages. To do this, it is sufficient to solve algorithmic problems using arrays and its capabilities in a programming language environment.

The convenience of programming in C ++ is that it is as versatile as any other programming language. This article will give you a good understanding of the structure of a C ++ program, the description of characters, the concept of algorithms and programs, the use of data input and output operators, as well as the ability to work with arrays, their types and rows in the program. It also provides general information about arrays, the order in which they are stored in memory, sorting, sorting, associating with functions, and a number of examples of their use in applications.

C ++ is a handy tool for creating programs that run in the Windows operating system environment, automating programming on a computer, reducing errors, and simplifying programmer work. The use of arrays in the C ++ programming language has several advantages over other programming languages. Working with arrays in C ++ makes some computational problems much easier. The speed of the program also increases several times due to their space in memory. This creates a number of conveniences for those who are familiar with the skills of programming languages in solving problems of technical specialization, and ensures a clear problem statement and high-precision solutions through appropriate algorithms.

REFERENCES:

1. A.A.Xaldjigitov, Sh.F.Madraximov, U.E.Adamboev. “Informatikavaprogrammalsh” O’quvqo’llanma. O’zMU, 2005 yil, 145 bet.
2. I. Qobulov “C++ tili“ Toshkentnash. 2008. – 114-b. https://tami.uz/matnga_qarang.php?id=1042

Editorial Board

Categories

- Business Management
- Social Science & Humanities
- Education
- Information Technology
- Scientific Fields

Review Process

Each research paper/article submitted to the journal is subject to the following reviewing process:

1. Each research paper/article will be initially evaluated by the editor to check the quality of the research article for the journal. The editor may make use of ithenticate/Viper software to examine the originality of research articles received.
2. The articles passed through screening at this level will be forwarded to two referees for blind peer review.
3. At this stage, two referees will carefully review the research article, each of whom will make a recommendation to publish the article in its present form/modify/reject.
4. The review process may take one/two months.
5. In case of acceptance of the article, journal reserves the right of making amendments in the final draft of the research paper to suit the journal's standard and requirement.

Published by

South Asian Academic Research Journals

A Publication of CDL College of Education, Jagadhri (Haryana)
(Affiliated to Kurukshetra University, Kurukshetra, India)

Our other publications :
South Asian Journal of Marketing &Management Research (SAJMMR)
ISSN (online) : 2249-877X
SAARJ Journal on Banking & Insurance Research (SJBIR)
ISSN (online) : 2319 – 1422