

Гущина Галина Владимировна

Государственное бюджетное общеобразовательное учреждение

средняя общеобразовательная школа №635

Приморского района Санкт-Петербурга

ОСНОВЫ ПРОГРАММИРОВАНИЯ

Тема **«Основы программирования»** входит в Государственный образовательный стандарт РФ.

Эта тема изучается в основной школе в 8-9 классе и в старшей школе при решении практических задач из различных предметных областей знаний.

Ознакомление с основами алгоритмизации и программирования входит в раздел курса по информатики и ИКТ «Программное обеспечение информационных технологий».

Цели обучения информатики и ИКТ в основной школе:

- формирование у учащегося системы базовых знаний по информатике;
- освоение базовой информационной технологии работы в системной среде Windows, в графическом редакторе, в текстовом процессоре, в табличном процессоре, в системе управления базой данных;
- освоение коммуникационной технологии в глобальной сети Интернет;
- формирование знаний по техническому обеспечению информационной технологии;
- приобретение знаний и умений целенаправленной работы с



информацией на основе системного подхода к анализу структуры объектов, создания и исследования информационных моделей;

- освоение информационной технологии моделирования в среде графического редактора, в текстовом процессоре;

- ознакомление с основами алгоритмизации и программирования;

- ознакомление с основами алгебры логики и логическими основами построения компьютера;

- понимание необходимости соблюдения этических и правовых норм информационной деятельности.

Цели изучения информатики и ИКТ в старшей школе направлено на достижение следующих целей:

- освоение системы базовых знаний, отражающих вклад информатики в формирование современной научной картины мира, роль информационных процессов в обществе, биологических и технических системах;

- овладение умениями применять, анализировать, преобразовывать информационные модели реальных объектов и процессов, используя при этом информационные и коммуникационные технологии, в том числе при изучении других школьных дисциплин;

- развитие познавательных интересов, интеллектуальных и творческих способностей путем освоения и использования методов информатики и средств ИКТ при изучении различных учебных предметов;

- воспитание ответственного отношения к соблюдению этических и правовых норм информационной деятельности;

- приобретение опыта использования информационных технологий в индивидуальной и коллективной учебной и познавательной, в том числе проектной деятельности.



Тема раздела «Основы алгоритмизации»

Понятие и определение алгоритма. Свойства алгоритмов. Формы представления алгоритма: словесная, графическая, табличная, программа. Типовые алгоритмические конструкции: последовательность, ветвление, цикл. Стадии создания алгоритма.

Линейный алгоритм. Разветвляющийся алгоритм. Циклический алгоритм. Цикл с известным числом повторений. Цикл с предусловием. Цикл с постусловием. Вспомогательный алгоритм.

Учащиеся должны знать:

- назначение алгоритма и его определение;
- свойства алгоритма;
- формы представления алгоритма;
- типовые алгоритмические конструкции;
- представление алгоритма в виде блок-схемы;
- основные стадии разработки алгоритма.

Учащиеся должны уметь:

- приводить примеры алгоритмов из разных сфер;
- составлять алгоритмы для различных ситуаций или процессов в виде блок-схем;
- разрабатывать циклические алгоритмы на основе различных видов ЦИКЛОВ.



Тема раздела « Представление о программе, классификация программ»

Исполнитель алгоритма. Понятия программы и программирования. Назначение процедуры. Подходы к созданию программы процедурный, объектный. Классификация и характеристика программного обеспечения: системное; прикладное; инструментарий программирования. Роль программного обеспечения в организации работы компьютера.

Учащиеся должны знать:

- понятия программы и программного обеспечения;
- отличие программы от алгоритма;
- назначение системного программного обеспечения;
- назначение прикладного программного обеспечения;
- назначение инструментария программирования.

Учащиеся должны уметь:

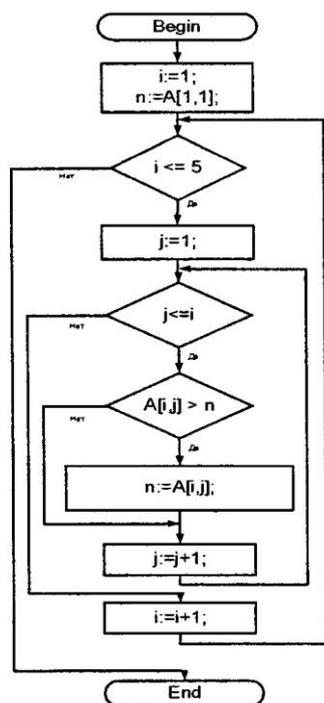
- классифицировать программы;
- объяснить различия процедурного и объектного подходов .при программировании па примерах из окружающей жизни.

Исходя из целей и задач, которые стоят перед учителем при изложении тем: *«Основы алгоритмизации»* и *« Представление о программе, классификация программ»* рассмотрим примеры задач. Здесь можно продемонстрировать решение наиболее характерных задач.



Задача 1.

Дана матрица. Найти значение максимального элемента матрицы среди элементов лежащих ниже главной диагонали матрицы, включая элементы главной диагонали. Записать программу и протестировать её для данной матрицы.



$$A = \begin{bmatrix} 12 & -2 & 7 & 26 & 0 \\ 0 & -5 & 4 & -18 & 32 \\ 22 & -1 & 3 & 0 & 28 \\ -9 & 24 & 2 & 25 & -15 \\ 2 & 18 & 11 & 4 & 1 \end{bmatrix}$$

Реализация алгоритма на языке C.

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

```
#include <stdlib.h>
```

```
int main()
```

```
{
```

```
const int A[5][5]={{ 12,-2,7,26,0},{0,-5,4,-18,32},{22,-1,3,0,28},{-9,24,2,25,-15},{2,18,11,4,1}};
```

```
int max_val,j,i;
```



```
printf("Matrix A:\n");
max_val=A[0][0];
for (i=0;i<5;i++)
{
for (j=0;j<=i;j++)
{
if (max_val<A[i][j])
max_val=A[i][j];
printf("%i ",A[i][j]);
}
printf("\n");
}

printf ("\nMax value is %i\n",max_val);

return 0;
}
```

Реализация алгоритма на C#.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace task1_1
{
```



```
class Program
```

```
{
```

```
static void Main(string[] args)
```

```
{
```

```
int[,] A = new int[5, 5] {{12,-2,7,26,0},{0,-5,4,-18,32},{22,-1,3,0,28},{-9,24,2,25,-15},{2,18,11,4,1}};
```

```
int max_val,j,i;
```

```
Console.WriteLine("Matrix A:\n");
```

```
max_val=A[0,0];
```

```
for (i=0;i<5;i++)
```

```
{
```

```
for (j=0;j<=i;j++)
```

```
{
```

```
if (max_val<A[i,j])
```

```
max_val=A[i,j];
```

```
Console.Write("{0} ", A[i,j]);
```

```
}
```

```
Console.Write("\n");
```

```
}
```

```
Console.Write("\nMax value is {0}\n", max_val);
```

```
Console.ReadKey();
```

```
}
```

```
}
```

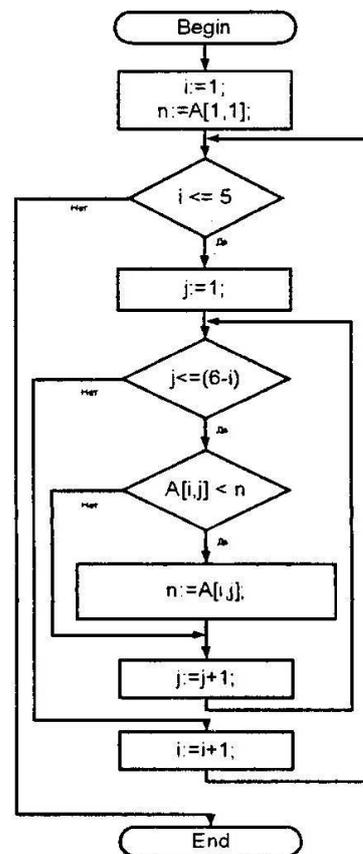
```
}
```



Задача 2.

Дана матрица. Найти значение минимального элемента матрицы среди элементов лежащих выше побочной диагонали матрицы, включая элементы этой диагонали. Записать программу и протестировать её для данной матрицы.

$$A = \begin{bmatrix} 3 & -2 & 7 & 6 & 0 \\ 0 & -5 & 4 & 18 & -8 \\ 4 & 11 & -6 & 0 & 28 \\ 9 & -7 & 2 & 5 & -15 \\ 2 & 18 & 11 & 4 & 1 \end{bmatrix}$$



Реализация алгоритма на языке C.

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

```
#include <stdlib.h>
```

```
int main()
```

```
{
```



```

const int A[5][5]={{12,-2,7,26,0},{0,-5,4,-18,32},{22,-1,3,0,28},{-9,24,2,25,-
15},{2,18,11,4,1}};

int max_val,j,i;

printf("Matrix A:\n");
max_val=A[0][0];
for (i=0;i<5;i++)
{
for (j=0;j<=i;j++)
{
if (max_val<A[i][j])
max_val=A[i][j];
printf("%i ",A[i][j]);
} //j
printf("\n");
} //i

printf ("\nMax value is %i\n",max_val);

return 0;
}

```

Реализация алгоритма на C#.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

```



```

namespace ConsoleApplication1
{
class Program
{
static void Main(string[] args)
{
int[,] A = new int[5, 5] {{3,-2,7,6,0},{0,-5,4,18,-8},{4,11,-6,0,28},{9,-7,2,5,-
15},{2,18,11,4,1}};
int min_val,j,i;

Console.WriteLine("Matrix A:\n");

min_val=A[0,0];
for (i=0;i<5;i++)
{
for (j=i;j<5;j++)
{
if (min_val>A[i,j])
min_val=A[i,j];
Console.Write("{0} ", A[i, j]);
}
Console.Write("\n");
}

Console.Write("\nMiin value is {0}\n", min_val);
Console.ReadKey();
}
}
}

```



Цели, преследуемые при решении первых двух задач:

- познакомить учащихся с общими принципами применения массивов и некоторыми базовыми приемами их обработки;
- научить определять двумерный массив;
- научить заполнять двумерный массив;
- уметь применять полученные знания при решении задач;
- способствовать развитию алгоритмического мышления учащихся.

Двумерный массив - это так называемая матрица (или таблица), у которой есть строки и столбцы. Принято, что первый индекс массива будет указывать на строки, а второй на столбцы. Вот пример объявления и инициализации двумерного массива, состоящего из пяти строк и пяти столбцов, реализованный в нашей задаче.

```
const int A[5][5]={{12,-2,7,26,0},{0,-5,4,-18,32},{22,-1,3,0,28},{-9,24,2,25,-15},{2,18,11,4,1}};
```

Двумерный массив имеет два индекса. Сразу при объявлении мы его инициализируем целочисленными величинами (int i,j).

Построим алгоритм поиска максимального элемента таким образом: введем дополнительную переменную с именем `max_val`. Присвоим начальное значение – переменной `max_val` значение первого элемента массива `A[0][0]`. Затем сравниваем каждый элемент с переменной `max`, если его значение оказывается больше, то присваиваем его переменной `max_val`.

Основные алгоритмы для обучения навыкам работы с двумерными массивами.

1. Заполнение двумерного массива.
2. Печать элементов массива.
3. Сумма элементов массива.
4. Вычисление суммы элементов каждой строки и каждого столбца.



5. Поиск максимального (минимального) элемента массива.
6. Поиск максимального (минимального) элемента массива и их индексов.
7. Работа с квадратными матрицами (вводится понятие главной диагонали матрицы): сумма элементов главной диагонали, сумма элементов выше (ниже) главной диагонали.

Задача 3.

Найти сумму двух чисел представленных в пятеричной системе счисления.

Провести тестирование программы для числа $a=4212_5$ и числа $b=2323$.

Сначала вспомним, как выполняется сложение в традиционной десятичной системе счисления.

Пусть надо сложить два числа: $748 + 254$.

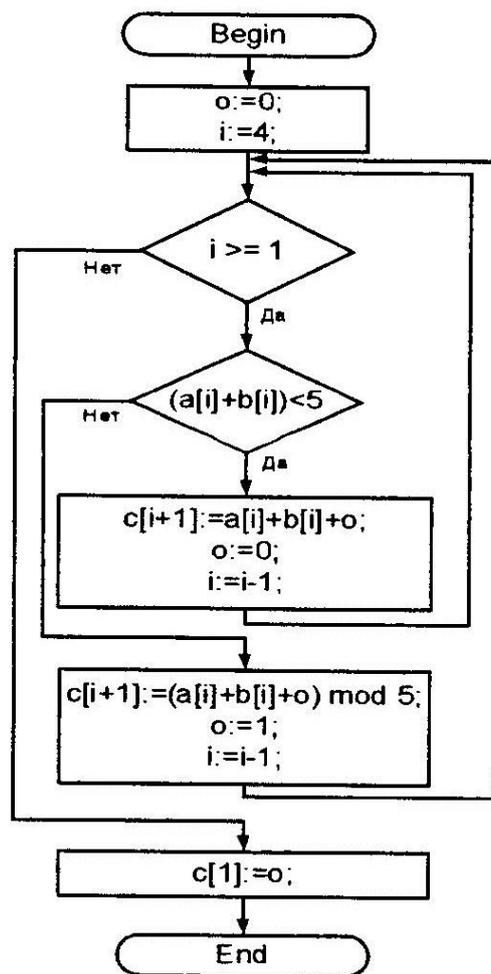
1. Складываем, начиная с самой правой позиции. $8+4=12$. Результат больше 10, поэтому переносим 1 в следующий слева разряд, вычитаем 10 из 12 и записываем в первой позиции 2. Переходим к следующей слева позиции.
2. Складываем 4 и 5. Получаем 9. Да еще была 1 из предыдущего разряда – всего 10. Результат равен 10. Вычитаем 10 и в данной позиции записываем 0. Переносим 1 в следующий разряд. Переходим к следующей слева позиции.
3. Складываем 7 и 2. Получаем 9. Да еще была 1 из предыдущего разряда – всего 10. Результат равен 10. Вычитаем 10 и в данной позиции записываем 0. Переносим 1 в следующий разряд. Переходим к следующей слева позиции.
4. Складывать уже нечего, но был перенос 1 из предыдущего разряда. Записываем окончательный результат: **1002**



Теперь сложим два числа в пятеричной системе счисления: $243_5 + 341_5$. Запишем сложение упрощенно. $3+1 = 4$. Переноса нет. $4+4 = 8$. Больше 5, значит 1 в перенос и $8-5 = 3$. $2+3+1$ (от переноса) = 6.

Больше 5, значит 1 в перенос и $6-5 = 1$. Есть 1 от переноса. Ответ: 1134_5 .

Алгоритм сложения двух чисел в пятеричной системе счисления представлен в блок-схеме.



Реализация алгоритма на языке C.

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

int main()
{
const int A[4]={4,2,1,2};
const int B[4]={2,3,2,3};
int C[5];

int z,i;

for (z=0;z<5;z++)
C[z]=0;

i=3;
for (z=0;z<4;z++)
{
C[z]=(C[z]+A[i]+B[i])%5;
if ((A[i]+B[i])>=5)
{
C[z+1]=C[z+1]+1;
}
i=i-1;
}

printf("Result value C=");
```



```
for (z=4;z>=0;z--)  
printf("%i",C[z]);
```

```
printf("\n");
```

```
return 0;
```

```
}
```

Реализация алгоритма на языке C#.

```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
namespace task3_1
```

```
{
```

```
class Program
```

```
{
```

```
static void Main(string[] args)
```

```
{
```

```
int[] A = new int[4] {4,2,1,2};
```

```
int[] B = new int[4] {2,3,2,3};
```

```
int[] C = new int[5];
```

```
int z,i;
```

```
for (z=0;z<5;z++)
```



```
C[z]=0;
```

```
i=3;
```

```
for (z=0;z<4;z++)
```

```
{
```

```
C[z]=(C[z]+A[i]+B[i])%5;
```

```
if ((A[i]+B[i])>=5)
```

```
{
```

```
C[z+1]=C[z+1]+1;
```

```
}
```

```
i=i-1;
```

```
}
```

```
Console.Write("Result value C=");
```

```
for (z=4;z>=0;z--)
```

```
Console.Write("{0}", C[z]);
```

```
Console.Write("\n");
```

```
Console.ReadKey();
```

```
}
```

```
}
```

```
}
```



Цели, преследуемые при решении третьей задачи:

- познакомить учащихся с позиционными системами счисления, представлением чисел в разных системах счисления.
- научить переводить числа из одной системы счисления в другую;
- научить складывать числа в разных системах счисления;
- уметь решать задачи на выполнение арифметических действий над числами, представленных в разных системах счисления;
- способствовать развитию алгоритмического мышления учащихся, повторение понятия алгоритма и требований к нему (результативность, однозначность, конечность, понятность).

Ссылки на интернет-ресурсы:

1. <http://iguania.ru/uchebnik-po-osnovam-programmirovaniya-dlya-nachinaiuschich/glava-15-dvumernie-massivi.html>
2. <http://festival.1september.ru/articles/517526/>
3. <http://server.179.ru/tasks/new-cpp/11-arrays.html>
4. <http://www.ict.edu.ru/ft/005667/terexov-k.pdf>
5. <http://www.education.gootkin.com/svodnaya>
6. http://www.oti.ru/forstudents/tasks/Alexey/obj494999/scale_of_notation.pdf

