**Практическое занятие 1.**

**Математические основы информатики.**

***Учебник - Павловская, Программирование на С++. Практикум***

*Цифра* – символ, знак.

*Число* – значение.

*Система счисления (СС)* – это комплекс разрешенных базисных знаков и правила образования из них любых чисел в данной системе счисления.

**Пример**: римская система счисления

**Правила:**

1. Если цифра справа меньше или равна цифре слева то эти цифры складываются.
2. Если цифра слева < цифры справа, то левая вычитается из правой

1 – I, 5 – V, 10 – X, 50 – L, 100 – C 500 – D, 1000 – M.

146 => CXLVI

*Виды СС*:

1. Аддитивные СС – любое число получается сложением или вычитанием базисных знаков (Римская СС)

ДОСТОИНСТВА

- Представление больших чисел небольшим количеством знака

НЕДОСТАТКИ

- Сложность проведения операций

1. Позиционные СС – значения каждого базисного знака изменяется от его позиции знаков составляющих это число

ДОСТОИНСТВА

- Легкость проведения операции

НЕДОСТАТКИ

- Большое количество знаков

*Основание СС* – количество разрешенных знаков в СС.

***Домашняя работа:*** перевести: 88, 90, 162 в римскую систему счисления.

**Домашняя работа**

88 = LXXXVIII

90 = XC

162 = CLXII

**Практическое занятие 2.**

**Форма представления числа в любой СС.**

1. В виде последовательности разрешенных базисных знаков.

Например: -1111.1111 где -1111 – это целая часть, 0.1111 - мантиса

1. В виде полинома

Например: 3\*10^2 + 4\*10^1 + 5\*10^0 + 6\*10^-1

**Действия в СС**

1. Сложение
2. Вычитание
3. Умножение

ДЗ:

10101 \* 1011

1001 \* 11

**Практическое занятие №3**

**Экономичность СС.**

Самыми экономичными из сравниваемых систем будет та СС, которая позволяет написать с помощью заданного количества знаков (одинаково для всех сравниваемых систем) наибольшее количество комбинаций

Пример:

Дано: 60 базисных знаков, 2-ная и 3-ная СС. Какая из них экономичней?

2-ная СС:

30 групп по 2 знака

Количество разных комбинаций: 2^30

3-ная СС:

20 групп по 3 знака

Количество комбинаций: 3^20

2^30 < 3^20

Ответ: 3-ная СС

ДЗ: дано 120 знаков 15 и 2 СС

Смешанные СС

Числа заданные в СС с основанием P переводятся в СС с основанием q где п меньше по таблице если основания связаны формулой P=q^l где l – количество разрядов в одном числе

Перевод из 10-ной в другие СС

Пусть п осн СС

1. Целое часть делится на п до получения целого частного, полученный при этом остаток в том числе и 0
2. Полученное частное опять делим на п до получения целого частного до тез
3. Полученные остатки будут разрядами числа в новой СС.
4. Число получается выписыванием в обратном порядке последнего частного и остатка

Алгоритм для мантисы

1. Мантиса последовательно умножается на основание P.
2. Полученные при этом целые части чисел являются очередным разрядом в новой СС
3. Умножение прекращаем по одной из двух причин 1. Либо когда 2. Либо когда будет достигнута заданная точность(обычно 4 числа после запятой)

ДЗ: 47,37 312.65 из 10сс в 3 другие

**Практическое занятие №4. Компьютерная система. Хранение информации в современных компьютерах. Числовая система компьютера.**

ДЗ: 235.875 в 10ной, 12345.672 в 8ной.

Числовая система компьютера напрямую связана с кодированием информации.

Кодирование - это формирование нового представления информации, то есть переход от исходного представления удобного для восприятия человеком, к представлению для хранения, передачи, и обработки информации. Декодирование – обратный процесс.

Цели кодирования информации.

1. Удобство физической реализации.
2. Удобство для восприятия человеком
3. Время передачи и обработки информационных кодированных и декодированных сообщений
4. Экономичность – снижения избыточности сообщения
5. Надежность – защита от случайных искажений
6. Защита от несанкционированного доступа

В зависимости от целей стоит выставлять системы приоритетов поскольку цели кодирования могут противоречить друг другу. Например: снижение экономичности может снизить

Уменьшение длинны сообщения может повысить сложность обработки

Объем хранимой информации при повышении может увеличить сложность обработки

Решение вопроса:

Выставление целей на разных этапах проекта

**Представление информации в компьютере.**

Обычно хранение информации происходит в виде двоичного кода

При этом искусственное количество памяти которой может быть размещено в одном элементе памяти называют битом

Который очень мал и сам по себе не несет никакой нагрузки

Поэтому для работы с информации предложено использовать более крупную единицу байт.

Байт - это минимальная область памяти которой присвоен адрес

Адрес – порядковый номер байта. Машинным словом называется такая ячейка памяти, которую аппаратная часть компьютера обрабатывает как единое целое. Длина машинного кода может быть различной. И именно она определяет архитектуру конкретного компьютера.

Точность вычислений зависит от архитектуры данного конкретного компьютера и длинны машинного кода

**Информатика от 02.10.2023**

**Практическое занятие №5**

**Преподаватель Полякова Ольга Андреевна**

Адресом машинного слова является адрес младшего байта этого слова

Объём информации, представляемой в конкретном компьютере, ограничен ёмкостью памяти данной ЭВМ, а, следовательно, числовая информация может быть представлена лишь с определенной степенью точности, которая зависит от архитектуры компьютера

Обмен информацией происходит через код ASCII т.е. код обмена, который генерируется внешними устройствами (например, принтером)

Дано:

RAM = 2kB

Какой адрес последнего байта оперативной памяти, если измерение начинается с нуля

2048 байт

Поскольку нумерация байтов начинается с нуля, адрес последнего байта будет равен 2047 (2048 –1)

Данные – это информация, представленная в виде пригодном для обработки автоматическими средствами

Дано:

RAM = 1мБ

Адрес последнего машинного слова равен 1048574

1мБ = 1024 кБ = 1048576 байт

Формат данных компьютера

1. Логические коды
2. Числа с фиксированной запятой
3. Числа с плавающей запятой

Логические коды

Логические коды могут быть размещены либо в отдельных байтах, либо в машинных словах, т.к. логическими кодами представлены: a) символьные величины b) Числа без знака c) битовые величины

Символьные величины – это определенные в кодах таблицы ASCII символы, где каждый из них занимает 1 байт, где 7-й бит всегда 0

В таблицы ASCII стандартны только первые 128 символов с адресами от 0 по 127. Сюда входят буквы латинского алфавита, арабские цифры, знаки препинания, скобки, служебные символы, остальные 128 символов (с 128 по 255) предназначены для кодирования национальных алфавитов, научных символов и других

Символ «I» в кодовой таблице имеет номер 105, определить, что зашифровано следующей последовательностью

108105110107 = L I N K

ABCDEFGHIJKNLMO

STOP = 115116111112

Числовая информация в компьютере

Числа с фиксированной запятой хранятся как целые числа

Числа с плавающей точкой хранятся как целые, так и вещественные числа

Числа с фиксированной запятой – целые числа

Диапазон представления таких чисел зависит от размера ячеек памяти т.е. от длины машинного слова. Например, в k-битовом машинном слове могу храниться 2k различных целых чисел

Дано:

Пусть в данном компьютере используется 16 разрядное машинное слово. Каков диапазон хранимых чисел, если

1. Используются только положительные числа
2. Используются числа положительные и отрицательные в равном к-ве
3. От 0 до 65535
4. От -32768 до 32767

ДЗ:

1. RAM имеет объём 4кБ, определить адрес последнего байта памяти
2. RAM содержит 163840 машинных слов. Это составляет 0.625мБ памяти. Какого длина машинного слова в битах
3. 4095 адрес последнего байта памяти
4. 32 бита

Представление целого положительного числа N

1. Перевести заданное число N В двоичную
2. Полученный результат дополним слева не значащими вариантами до K разрядного машинного слова
3. Вписываем результат машинного слова
4. Представляем упакованную форму числа в двоичной системе

Дано:

N = 1607 в дес. Системе

N = 11001000111

11 разрядов – 2 байта

0000 0110 0100 0111

0000 0110 | 0100 0111

0 6 4 7

0647

**Представление в компьютере целого отрицательного числа**

1. **Получаем компьютерного представления числа со знаком +**
2. **Получаем обратный код числа заменив 0 на 1 а 1 на 0**
3. **Получаем дополнительный код числа добовляя едницу в посл разряд обратного кода**
4. **Получаем в 16чной СС**

N = 0000 0110 0100 0111

1111 1001 1011 1000 – обратный код

1111 1001 1011 1001 – дополнительный код

F 9 B 9

Ответ: F9B9

- 195 , -263, -3120

**Представление вещественных или действительных чисел привязаны к след формуле**

R = m\*n^p

Где m – число; n основание СС; р – порядок числа

Дано:

25.324 = 2.5324 \* 10^1 = 2532.4 \* 10^-2 = **0.25324 \* 10^2**

**В памяти хранятся –**

1. Знак числа
2. Цифры мантиссы
3. Порядок числа (степень)
4. Знак порядка

Пусть дано K-разрядное машинное слово, где K равно 32. В компьютере действительные числа хранятся в нормализованной системе с плавающей запятой, где мантиса меньше единицы и не 0, а формат представления соответствует формуле 1

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2-7 | 8 | - | 32 |

1. Знак числа (1 - ; 0 +)

2-7. Машинный порядок числа

Min = 000 0000 = 0 dec.

Max = 111 1111 = 127 dec.

128 машинных порядков – 64 положительных, 64 отрицательных. (-64 – 63)

8-32 – Мантисса

В компьютере математические машинные порядки сдвинуты относительно друг друга так, чтобы начальному значению соответствовал 0, следовательно машинный порядок сдвинут на 64 единиц вправо

Mp = p + 1000000 bin.

1. Переводим модуль заданного действительного числа в двоичную СС с 24 значащими цифрами (если мантисса короче, то справа добавляем 0)
2. Нормализуем полученное число по формуле 1 представляя его как 0. 24 цифры)
3. Вычисляем порядок Мр
4. Размещаем число в машинное слово
5. Переводим в 16-ную СС

Дано:

R = 250.1875 dec.

1111 1010.0011 0000 0000 0000 bin.

0.1111 1010 0011 0000 0000 0000 bin \* 10^1000 bin

Mp = 1000 + 100 0000 = 1001000

|  |  |  |  |
| --- | --- | --- | --- |
| 0|100 1000 | 1111 1010 | 0011 0000 | 0000 0000 |

4 8 F A 3 0 0 0

Ответ:48FA3000

Дз 501.75

**Практическое занятие №5. Измерение информации.**

Человек всегда хотел получить цифровое измерение следующих составляющих:

1. Материя – измерение вещественного, то есть количество вещества, которое определяется весом глубиной высотой объемом площадью и другие измерения
2. Энергия – определяется ее количеством, где единица измерения привязаны к ее виду: тепловая, механическая, электрическая, атомная и другие виды (Ваты, джоули, ньютоны килокалории)
3. Информация –

Дано:

* Энергия взмаха крыла бабочки 10^-4
* Пуля ружья 10^2
* Атомная бомба 10^20

Дано:

* Яблоко 200 гр.
* Атом 10 ^-20
* Планета 6\*10 ^27

Бит это количество информации которая уменьшает неопределенность информации об исследуемом объекте ровно в 2 раза

Количество возможных результатов называется неопределенность нашего знания.

Формула Хартли (Для равновероятных событий)

* N=2^i
* I = log2(N)
* N – количество возможных исходов
* I – количество информации в битах для данного события

Дано:

В барабане 32 шара. Сколько информации несет в себе информация о выпадении 15. От

Ответ 5бит

Дано

* Кубик – 6 граней
* 2.59 бит

Количество информации и вероятность

Дано в коробке 50 шаров. 40 белых. Найти вероятность выпадения белых (7/8)

Чем меньше вероятность события, тем больше информации оно содержит.

Для неравновероятных событий используется формула Шеннона.

Дано: В корзине белые и черные шары, из них 18 черных. Сообщение о том что достали белый шар несет 2 бита информации. Сколько белых шаров

1. Пусть X количество шаров
2. I бел. – 2 бита
3. По формуле Хартли **I=log2 N**
4. I = log2(1p) = 2 log2(1p.б) = ¼
5. Т. К. вероятность выпадения б = ¼ то ч = ¾
6. ¾ = 18/x
7. X = 24

Модель данных – соглашение, в рамках которого конкретная реализация компилятора (интерпретатора) языка программирования определяет размер (в байтах/битах) фундаментальных (базовых) типов данных

Модель данных может кодироваться 3-мя значениями: размером целого числа (int), размером указателя (prt)

**32 битные**

Максимальное ограничение адресного пространства в 32-битных моделях равно 4 Гигабайтам

**Расшифровка названия – Long, Pointer 32.**

Использовалась на 32-битных системах, преймущественно в Win16 API.

**ILP32 (4/4/4)**

Расшифровка названии: Integer, Long, Pointer 32.

* Win32
* 32битные Unix и Unix – подобные системы (включая Limux и MacOS (OS X))

**64-битные**

**LLP64 (4/4/8)**

Int, Long – 32, Pointer – 64

**LP64 (4/8/8)**

Int – 32, Long – 64, Pointer – 64

64-битные Unix и Unix-подобные системы (включая Linux и MacOS)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип данных | Стандарт С++ | LP32 | ILP32 | LLP64 | LP64 |
| Указатель (pointer) | Не менее 32 | 32 | 32 | 64 | 64 |
| Короткое целое (Short int) | Не менее 16 | 16 | 16 | 16 | 16 |
| Целое | Не менее 16 | 16 | 32 | 32 | 32 |
| Длинное целое | Не менее 32 | 32 | 32 | 32 | 64 |
| Длинное длинное целое | Не менее 64 | 64 | 64 | 64 | 64 |
| Плавающая точка одинарной точности | Не менее 32 | 32 | 32 | 32 | 32 |
| Плавающая точка двойной точности | Не менее 64 | 64 | 64 | 64 | 64 |

Реализация float и double подчиняется стандарту IEEE 754

**Для float:**

1. Знаковый бит: 1
2. Порядок 8
3. Мантисса: 23 (вычисляется до 24 знака, последний бит явно не хранится)

**Для double:**

1. Знаковый бит: 1
2. Порядок: 11
3. Мантисса: 52 (Вычисляется до 53 знака, последний бит явно не хранится)

**Формула Шеннона**

I = log2(1/p) где p – вероятность наступления события

P = 4/5 красные 0.8 Log(1.25) = 0.32 бита

P = 1/5 зеленый 0.2 Log(5) = 2.32 бита

Алфавитный подход к измерению информации

Алфавитный подход связан с наукой – кибернетика.

Кибернетика - это наука, которая интересуется сложными объектами в том числе и живыми организмами но изучает не внутреннее содержание объекта а процессы взаимодействия объектов, их связях, и управление ими.

В кибернетике моделью представление сложных объектов является черный ящик где главными характеристиками является входные и выходные данные.

Несколько черных ящиков обмениваются входной и выходной информацией которая является последовательностью сигналов. Сигнал представляет собой некую символьную форму, которая называется символьная форма информации

Дано Русский алфавит 54 символа.

Формула Хартли для символов: I = log(N)

Где I количество информации символа

N – Количество символов в алфавите(мощность)

* 1. бит информации несет в себе символ

Дано Английский алфавит содержит 44

5.555 бит

Количество информации зависит от мощности алфавита и размера текста.

Основы математической логики

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| a | b | Инверсия (!) не (-a) | Конъюнкция (&&) и | Дизъюнкция (||) или | Импликация  (->) a <= b | Эквивалентность  (⬄) a==b |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 1 |

**Введение в теорию алгоритмов**

Алгоритм – Это последовательность действий, которые ведут от исходных данных к нужному результату. Новый набор данных определяет новое протекание процессов.

Алгоритм имеет 3 свойства:

1. Детерминированность (определенность) – каждое действие алгоритма понятна исполнителю, иначе должен быть предусмотрен аварийный останов.
2. Массовость – способность алгоритма решать не одну задачу, а целый класс задач, при этом одна задача от другой отличается исходными значениями данных
3. Конечность – гарантированное получение результата за конечное время.

Формы представления алгоритмов (т=м

1. Словесное описание алгоритма
   1. Пример:

Программа перехода

Если желтый мигает или зеленый

Перейти убедившись в том что тебя пропускает автомобили

В противном случае

Стоять до смены сигнала светофора

Алгоритм применять при переходе улицы со светофором

Переход 1

Пройти прямо

Переход 2

1. Блок схемы
2. Код выполнения на алгоритмическом языке

Требования

1. Простота – включение в алгоритм действительно необходимых элементов и их связей
2. Универсальность – модель должна реализовать любой алгоритм данного класса

Классы моделей алгоритмов

1. Арифметические алгоритмы – основаны на правилах:
   1. Любые данные должны быть закодированы числами, а их преобразование должны быть представлены вычислениями

Такой алгоритм - это числовая функция, где каждый шаг – это операция

Способы реализации:

1. Подставление функции в функцию – способ суперпозиции
2. Рекурсия – функция вызывает саму себя
3. Такой класс что для реализации понимания алгоритмов нужна такая машина чтобы каждый шаг выполнялся элементарно в 1 действие и к действиям этой машины предъявляются требования простоты и универсальности

**Абстрактная машина Тьюринга**

Машина Тьюринга определяет

* вычислимость функции
* Решает проблему определения: можно ли создать алгоритм который определит будет ли работать другой алгоритм (будет ли он завершенным при новом наборе входных данных)

Структура:

1. Бесконечная лента
2. Голова машины
3. Набор состояний (управляющее воздействие)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| < > | Ai | aj | ak | ….. |  |  |  |  |  |

^ - голов машины

| Qi | - устройство управления

Q = {qi … qn}

A = {ai…..an}

Шаг команды qi a1 => qj a2 – перемещение на один шаг влево и на один шаг вправо или остановка

Является ли строка палиндропом

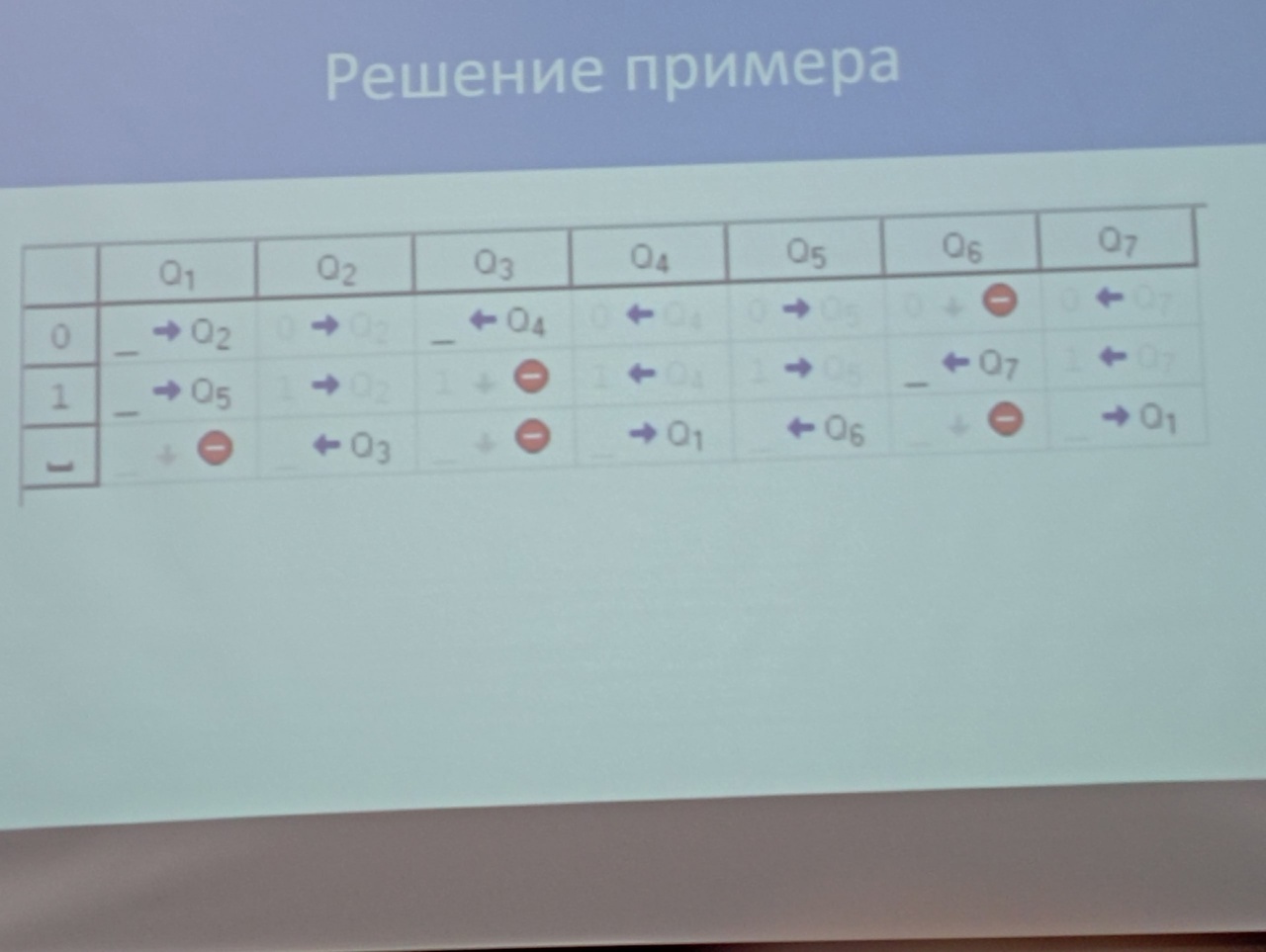
А = 0,1

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 0 | 1 | 0 | 1 |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 |  | < > | 0 | 1 | 0 | 1 |  |  |  |
| 1 = 1 |  | < > | 0 | 1 | 0 | < > |  |  |  |
| 0 |  | < > | < > | 1 | 0 | < > |  |  |  |
| 0 = 0 |  | < > | < > | 1 | < > | < > |  |  |  |
| 1 |  | < > | < > | < > | < > | < > |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

Если не палиндром то строка будет содержать символы

**Алгоритм оформляется в виде таблицы состояния машины тьюринга**



Q1 -считывание символа слева

Q2 – запоминание первого символа если он = 0, замена на пустой символ, и перемещает вправо до 1го пустой ячейки

Q3 – возврат на 1 символ влево

Q4 – перемещение влево до 1й пустой ячейки

Q5 – запоминание первого символа если он = 1, замена на пустой символ, и перемещение в право

Q6 – проверка последнего символа на равность 1

Q7 – сдвиг влево до первого пустого символа

Q0 – Остановка алгоритма

Полностью пнипу

Лаб раб машина Тьюринга

Выполнил: студент группы РИС-23-1б

ФИО

Проверила доцент кафедры ИТАС:

О. А . Полякова

2023 г.

Разработка алгоритма работы машины Тьюринга

1. Постановка задачи
2. Словесный алгоритм – как решать, - смысловое значение команд
3. Таблица команд
4. Разбор задачи на ленте пошагово
5. Результаты работ - скриншоты
   1. Разные данные пропускаем через движок
   2. Скины гитхаба
6. Вывод
7. Алфавит – множество символов которое задано
8. Правила подстановки просматриваются сверху вниз
9. Первым применяется то правило, которое позволяет в заданном слове (слово читается слева на право) отыскать композицию из символов одинаковую слева частью этого правила.
10. Заменяем правую часть этого правила
11. Как только применилось одно правило, далее преобразование строки начинаем с применением первого правила
12. Если ни одно правило применить не возможно, то преобразование заканчивается

Для избегания зацикливания алгоритма Маркова, применяются терминальные правила, после выполнения которого алгоритм прекращается (|->).

1. Нетерминальное правило – преобразования продолжаются (->)

|->

**Семинар 8. Обработка программы на C++.**

**Обработка.**

Компиляция – процесс преобразования исходного кода в машинный код

Происходит в 4 этапа:

1. Работа препроцессора (препроцессирование)
2. Трансляция (перевод в ассемблер)
3. Ассемблирование (перевод из кода ассемблера в машинный код)
4. Компановка (перевод

Текстовый редактор формирует исходный файл, и сохраняет в расширении .срр.

Задача – упрощать и не допускать возможные ошибки. В результате получаем исходный набор файлов. В с++ программа - это набор функций.

Далее в файл после обработки текстовым редактором, включаются файлы заголовков (хэдеры) с расширением .h

Препроцессор отмечает точки где будут происходить замены, на действия (директивы), и дает возможность запуска.

Директивы начинаются с #. Например: #include <iostream>

Где include – действие (директивы)

#Define – обычно используется для определения макросов связанных с константами, ключевыми словами или часто используемыми операторами или выражениями

#If, #elif #endif #else

#Error – выдает указанное пользователем сообщение об ошибке

#Include – включение библиотеки

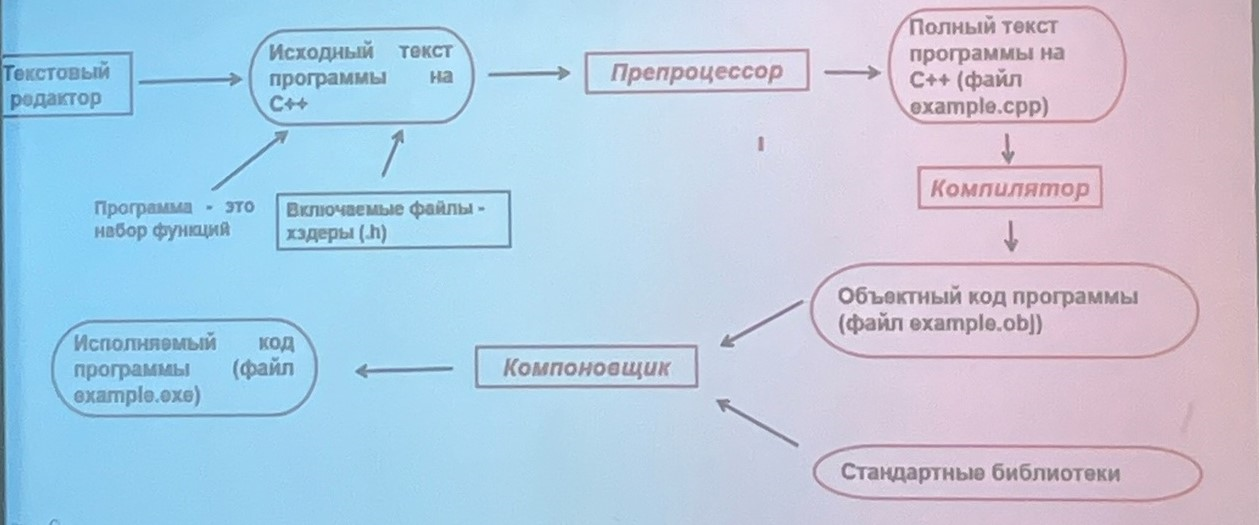
#Pragma – исключает проблему повторного включения

<iostream> - библиотека

Теперь получаем полный текст программы расширения .срр

После компилирования получаем объектный файл .obj на ассемблере.

Компоновщик (линкер) связывает все обьектные файлы и статистические библиотеки в единый файл .exe



**Пространства имен**

Пространства имен служат для логического группирования объявлений и ограничения доступа к ним.

Namespace имя пространства {}

Using директива подключается с целью увеличения читабельности и улучшения производительности.

Using namespace имя пространства;

Using имя пространства :: имя элемента;

После объявления пространства мы можем обращаться к его элементам

Имя пространства :: Имя элемента

Пример: MATH::Pi возвращает значение Pi

Локализация – процесс адаптации продукта к конкретному региону и времени.

Setlocale(LC\_ALL, “Russian”);

**ДЗ СПД 19.701 90 – Распечатать**