

Языки программирования

Генерация кода

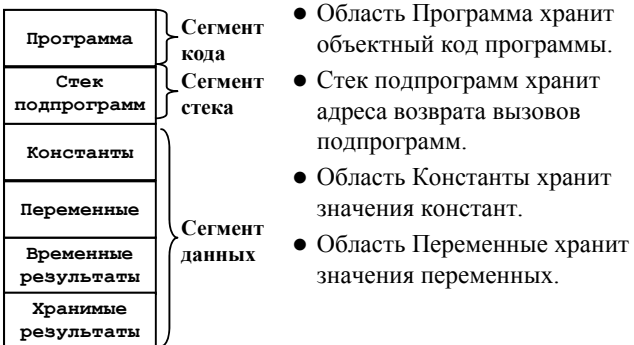
Содержание

- Распределение памяти
- Организация таблиц
- Подпрограмма генерации кода
- Методы оптимизации кода

Языки программирования

© М.Л. Цымблер

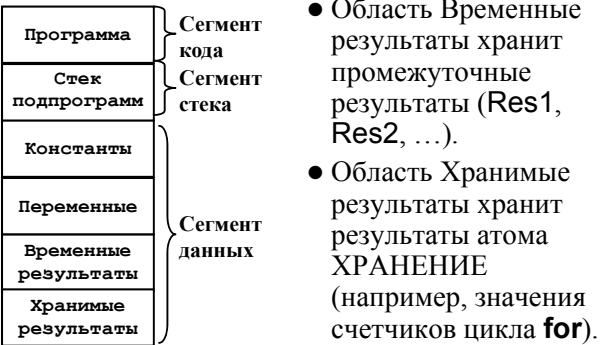
Распределение памяти



Языки программирования

© М.Л. Цымблер

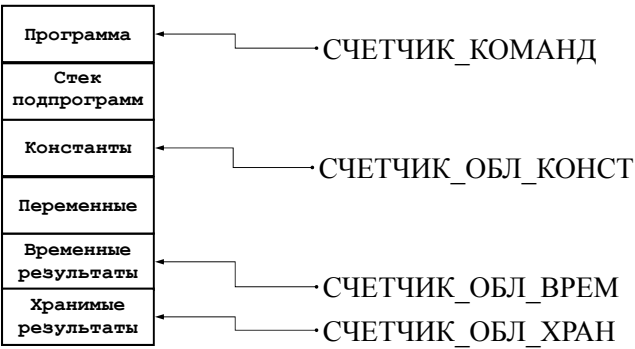
Распределение памяти



Языки программирования

© М.Л. Цымблер

Переменные распределения памяти



Языки программирования

© М.Л. Цымблер

Константы распределения памяти



Языки программирования

© М.Л. Цымблер

Организация таблиц

- *Таблица констант* содержит значения констант.
 - Заполняется лексическим блоком.
- *Таблица переменных* содержит адреса переменных в период исполнения программы.
 - Адреса назначаются разработчиком компилятора.
- *Таблица хранимых результатов* содержит адреса результатов атома ХРАНЕНИЕ (например, значения счетчиков цикла **for**).
 - Заполняется путем вызова подпрограммы **НОВТХРАН**, которая возвращает текущий свободный адрес в области хранимых результатов, увеличивает значение переменной **СЧЕТЧИК_ОБЛ_ХРАН** и сравнивает его с константой **ГРАНИЦА_ОБЛ_ХРАН**.
- *Таблица временных результатов* хранит адреса промежуточных результатов во время исполнения программы.
 - Адреса промежуточных результатов динамически изменяются в процессе компиляции. Нулевой адрес показывает, что в период исполнения результат находится в регистре (например, сумматоре).

Языки программирования

© М.Л. Цымблер

Подпрограмма генерации кода

- **ГЕН_КОД**(код_операции, указатель_на_элемент_таблицы)
- Выполняет следующие действия:
 1. Формирует двоичный код для операции с кодом **код_операции**.
 2. Формирует двоичный код для элемента таблицы с адресом **указатель_на_элемент_таблицы** в зависимости от вида элемента.
 3. Помещает двоичный код команды по адресу **СЧЕТЧИК_КОМАНД**.
 4. **СЧЕТЧИК_КОМАНД := СЧЕТЧИК_КОМАНД + 1**

Языки программирования

© М.Л. Цымблер

Методы оптимизации кода

- *Использование регистров* вместо обычных ячеек.
- *Оптимизация атома* – генерация более эффективных команд для частных случаев.
- *Оптимизация оператора* – переупорядочение атомов оператора.
- *Оптимизация нескольких операторов* – однократное вычисление общих подвыражений и др.
- *Оптимизация циклов*.
- *Замена операций* на более быстрые.
- *Удаление бесполезных операторов*.

Языки программирования

© М.Л. Цымблер

Оптимизация атома

● СЛОЖ(A,B,C)

- если переменная $B=0$, то данный атом можно заменить на ПРИСВ(C,A)
- если A и B – константы, то данный атом можно заменить на ПРИСВ(C,R), где $R=A+B$

● ПРИСВ(C,A)

- если переменная $A=0$, то данный атом можно заменить на ОБНУЛ(C)

● УМНОЖ(A,A,C)

- данный атом можно заменить на КВАДРАТ(A,C)

Языки программирования

© М.Л. Цымблер

Оптимизация оператора

● Переупорядочение выражений

Пусть для вычисления выражения $A*B+C*D$ (польская запись $AB*CD*+$) имеется только один регистр.

№ п/п	Обычный код	Оптимизированный код
1	MOV(Регистр_Сумматор,A)	MOV(Регистр_Сумматор,C)
2	MUL(Регистр_Сумматор,B)	MUL(Регистр_Сумматор,D)
3	MOV(Рез1,Регистр_Сумматор)	MOV(Рез1,Регистр_Сумматор)
4	MOV(Регистр_Сумматор,C)	MOV(Регистр_Сумматор,A)
5	MUL(Регистр_Сумматор,D)	MUL(Регистр_Сумматор,B)
6	MOV(Рез2,Регистр_Сумматор)	ADD(Регистр_Сумматор,Рез1)
7	MOV(Регистр_Сумматор,Рез1)	
8	MUL(Регистр_Сумматор,Рез2)	

Языки программирования

© М.Л. Цымблер

Оптимизация оператора

● Оптимизация вызова подпрограмм

- Для вызова процедуры `procedure Dummy(var A: Real); begin end;` никакой объектный код не генерируется.
- Для вызова процедуры `procedure Fake; begin A:=53; end;` генерируется код для оператора `A:=53;` (команда вызова подпрограммы не генерируется).

Языки программирования

© М.Л. Цымблер

Оптимизация нескольких операторов

- *Однократное вычисление общих подвыражений*
 - Для последовательности операторов
 $K:=(A+B)*C-D/(A+B);$
 $L:=\text{Sqrt}(A+B);$
 $Z:=A+B-L;$
 можно генерировать код, вычисляющий $A+B$ только один раз
- *Выполнение константных операторов*
 - Для последовательности операторов
 $A:=1; B:=2; C:=3;$
 $D:=A+B+C;$
 можно генерировать код для оператора $D:=6$

Языки программирования

© М.Л. Цымблер

Оптимизация циклов

Метод	До оптимизации	Оптимизированный код
<i>«Чистка» цикла (выделение инвариантов)</i>	for i:=1 to 10000 do begin A[i]:=(C+D)*i; K:=N-M; end;	Z:=C+D; K:=N-M; for i:=1 to 10000 do A[i]:=Z*i;
<i>Слияние циклов</i>	for i:=1 to 10000 do A[i]:=1; for j:=1 to 10000 do B[j]:=E*C[j];	for k:=1 to 10000 do begin A[k]:=1; B[k]:=E*C[k]; end;
<i>Разбиение циклов</i>	for i:=1 to 10000 do if X>Y then A[i]:=B[i]+X else A[i]:=B[i]-Y;	if X>Y then for i:=1 to 10000 do A[i]:=B[i]+X else for i:=1 to 10000 do A[i]:=B[i]-Y;

Языки программирования

© М.Л. Цымблер

Оптимизация циклов

Метод	До оптимизации	Оптимизированный код
<i>Замена операций в теле цикла</i>	for i:=1 to 10000 do A[i]:=i*5;	j:=5; for i:=1 to 10000 do begin A[i]:=j; j:=j+5; end;
<i>Развертывание циклов</i>	for (i=0; i<10000; i++) A[i]:=B[i]*C[i];	for (i=0; i<10000; i+=2) { A[i]:=B[i]*C[i]; A[i+1]:=B[i+1]*C[i+1]; }

Языки программирования

© М.Л. Цымблер

Замена операций

- *Использование инкрементных и декрементных операций* вместо сложения и вычитания
 - Inc(A) { генерируется для A:=A+1 }
 - Dec(A) { генерируется для A:=A-1 }
- *Использование операций сдвига* вместо целочисленного умножения и деления
 - A:=B shl 3 { генерируется для A:=B*8 }
 - A:=B shr 2 { генерируется для A:=B div 4 }

Языки программирования

© М.Л. Цымблер

Удаление бесполезных операторов

- Цепочку операторов
A:=1; A:=2; A:=3;
можно заменить на оператор A:=3;
- Оператор
if A>1 then B:=2 else ;
можно заменить на оператор
if A>1 then B:=2;
- Операторы и программные объекты исходного кода, которым никогда не передается управление, могут не включаться в объектный код.

Языки программирования

© М.Л. Цымблер

Заключение

- Оперативная память в период исполнения программы состоит из **сегмента кода, сегмента стека и сегмента данных** (разделенного на **область констант, область переменных, область временных результатов и область хранимых результатов**).
- Для хранения информации о параметрах атомов генератор кода использует следующие таблицы: **таблица констант, таблица переменных, таблица хранимых результатов и таблица временных результатов**.

Языки программирования

© М.Л. Цымблер

Заключение

- Для генерации кода используется специальная подпрограмма **ГЕН_КОД**, в том числе выполняющая изменение **счетчика команд**.
- **Эффективность объектного кода** может быть повышена с помощью различных приемов, например: распределение регистров, оптимизация атомов, операторов и циклов замена операций, удаление бесполезных операторов и др.
