Лекция 8

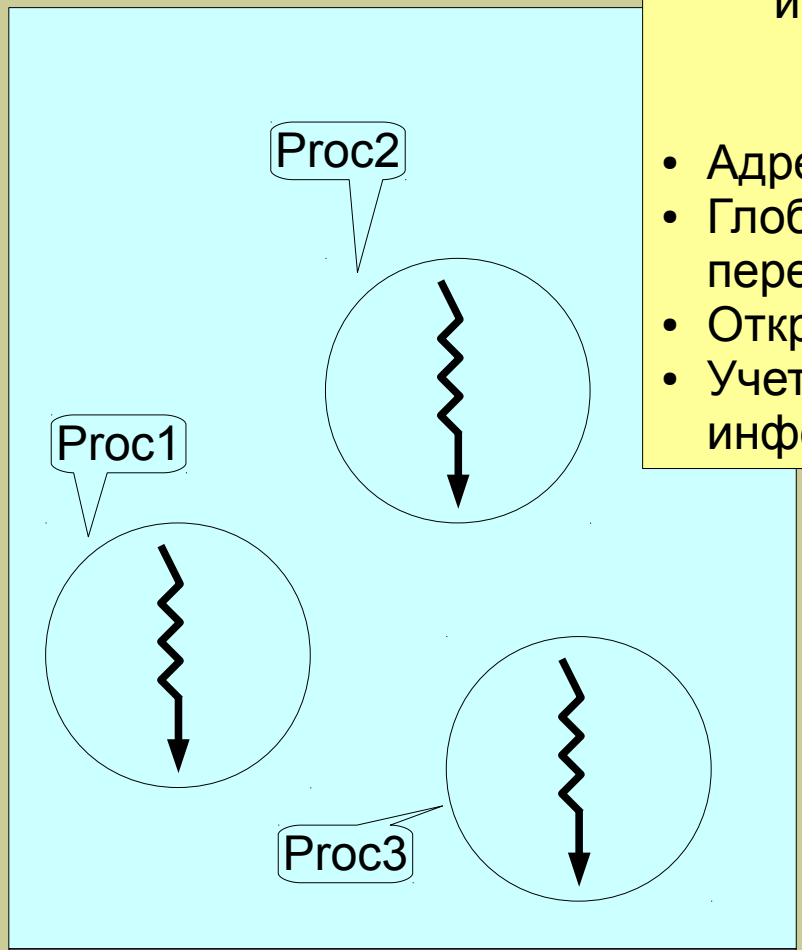
Потоки

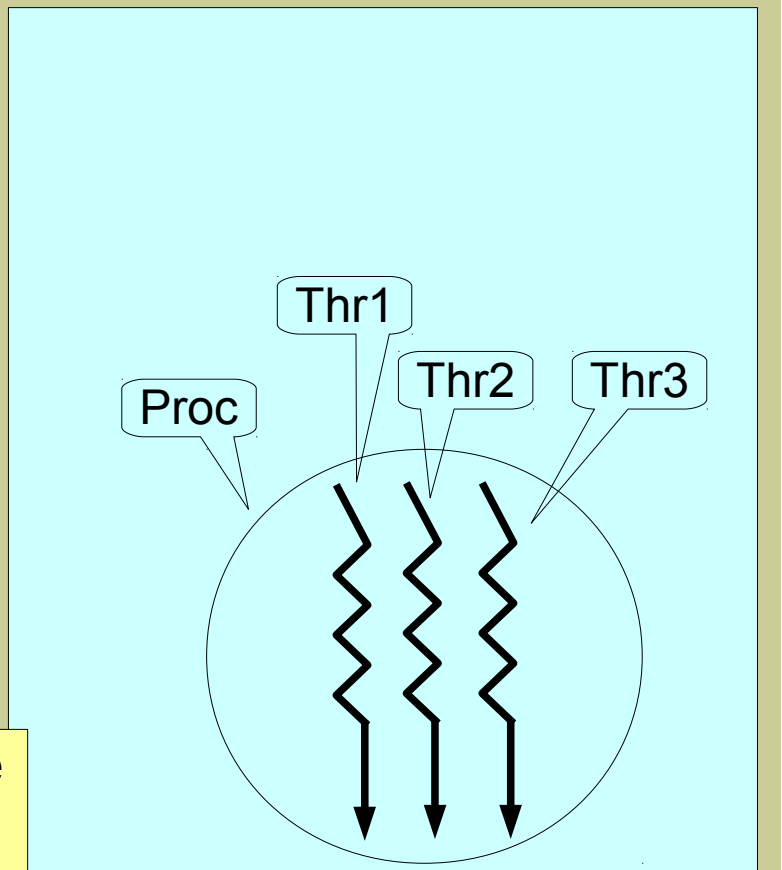
I. Потоки выполнения: применение - многопоточное программирование, модель потоков, объект ядра.

II. Реализация потоков: Visual C++, STL C++11, CreateThread.

Совместно используемые ресурсы

- Адресное пр-во
- Глобальные переменные
- Открытые файлы
- Учетная информация





Индивидуальные элементы потоков

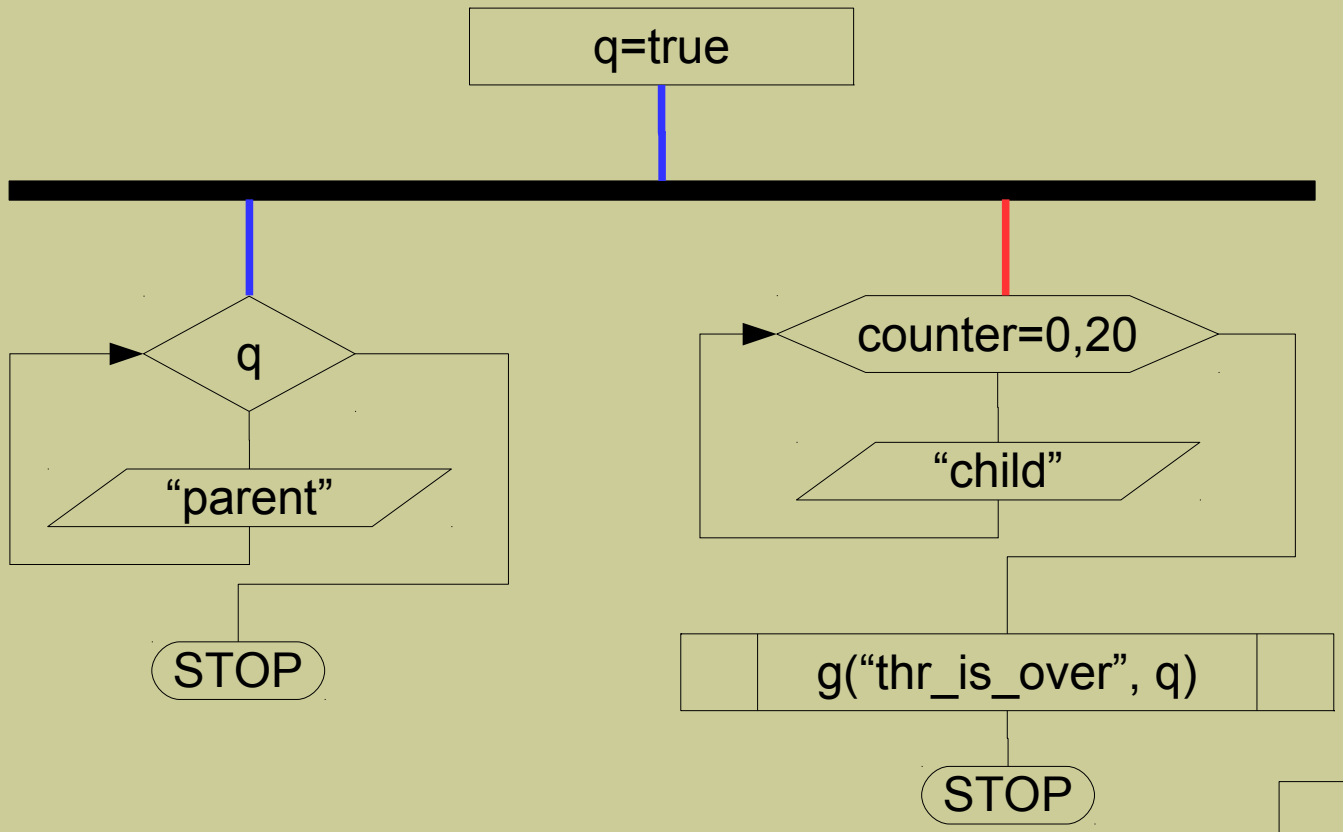
- Счетчик команд
- Регистры
- Стек
- Состояние

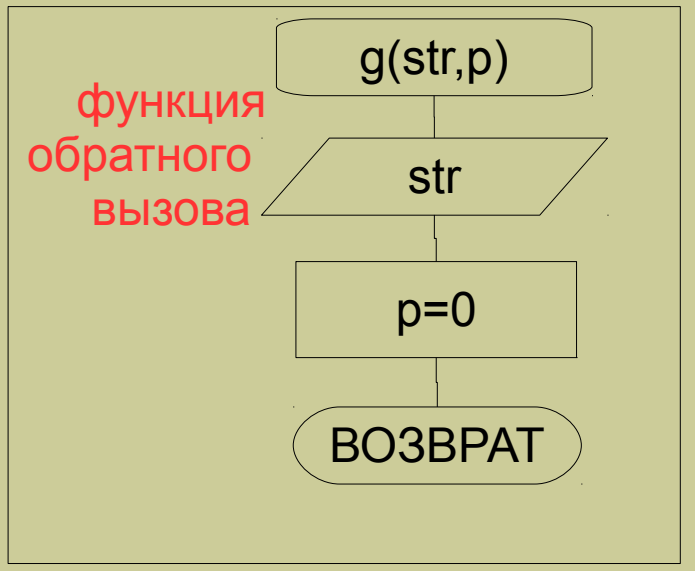
Таблица процессов

Таблица процессов

Таблица потоков

Многопоточная программа





Реализация Visual C++

(C Run-Time Library) (libcmt.lib; /MT, _MT)

unsigned long _beginthread(void(__cdecl ****start_address* **)(void *),
unsigned *stack_size*, **void ****arglist* **);**

void _endthread(void);

Реализация C++11

```
#include <thread> //класс thread
……..
void fun(int, double); //тип void необязателен
……...
std::thread  test(fun, par1,par2,…); //конструктор
……..
test.join();  // синхронизация
```

Реализация Visual C++

```
//  > cl /MT /D "_X86_" th4.c
 #include <windows.h>
 #include <process.h>
 #include <stdio.h>

 typedef int (*fun)(char*,int* p);

 int g(char* str, int* p){
    *p=0;
     printf("%s\n",str);
     return 0;
 }

 int q=1;
```

Реализация C++11

```
//  > cl /MT /D "_X86_" th4.c
 #include <windows.h>
 #include <thread>
 #include <stdio.h>

 typedef int (*fun)(char*,int* p);

 int g(char* str, int* p){
    *p=0;
     printf("%s\n",str);
     return 0;
 }

 int q=1;
```

Реализация Win32 API

```
//  > cl /MT /D "_X86_" th4.c
 #include <windows.h>

 #include <stdio.h>

 typedef int (*fun)(char*,int* p);

 int g(char* str, int* p){
    *p=0;
      printf("%s\n",str);
      return 0;
 }

 int q=1;
```

Реализация Visual C++ и C++11

```
void Thread( void* pg ) {
   int counter = 0;
   while ( q ){
        printf("child\n");
        Sleep(10);
            if(counter++ > 1000)
                                    break;
    }
        ((fun)pg)("thread_is_over!",&q);
}
```

Реализация Win32 API

```
DWORD WINAPI  Thread( void* pg ) {
   int counter = 0;
   while ( q ){
        printf("child\n");
        Sleep(10);
             if(counter++ > 1000)
                                  break;
    }
    ((fun)pg)("thread_is_over!",&q);
    return 0;
}
```

Реализация Visual C++

```
int main( void ){
  _beginthread( Thread, 0, (void*)g );
    while(q){
            printf("parent\n", q);
            Sleep(10);
    }

   return 0;
}
```

Реализация C++11

```
int main( void ){
    std::thread thr(Thread,(void*)g);
    while(q){
            printf("parent\n", q);
            Sleep(10);
    }
    thr.join();
    return 0;
}
```

Реализация Win32 API

```
int main( void ){
    DWORD dwThreadId;
    CreateThread(
        NULL,          // аттрибуты безопасности по умолчанию
        0,             // размер стека  потока по умолчанию
        Thread,        // функция потока
        g,             // аргумент функции потока
        0,             // флаги создания потока по умолчанию
        &dwThreadId);  // возврат идентификатора потока

    while(q){
      printf("parent\n", q);
      Sleep(10);
    }
    return 0;
}
```