C++

exceptions, rtti, templates

Исключения 1

Синтаксис:

```
try {
}
catch (тип имя) {
}
catch (тип) {
}
catch(…) {
}
```

Порождение:

```
throw [выражение];
```

Исключения 2

Пример:

```
void* malloc(size_t n) {
    void* p = HeapAlloc(GetProcessHeap(), 0, n);
    if(!p) throw “No memory”;
    return p;
}
void func {
    char* p1 = (char*)malloc(100);
    char* p2 = (char*)malloc(100);
    delete p2;
    delete p1;
}
try {
    func();
} catch(const char* p) {
    printf(“Error: %s”, p);
}
```

Преобразования типов

Синтаксис:

1) тип (выражение);

2) (тип) выражение;

Примеры:

```
const char* p = “asdf”; int i; char* a;
1) i = p;
2) i = (int)p;
3) i = int(p);
4) a = p;
5) a = (char*)p;
6) a = char*(p);
```

const_cast

Синтаксис:

const_cast <тип> (выражение);

Выражение: константный указатель

Тип: тот же что и тип выражения но без const

Возвращаемое значение: переменую типа ‘тип’

Пример:

```
const char* p = “asdf”;
char* a = const_cast<char*>(p);
```

dynamic_cast 1

Синтаксис:

dynamic_cast <тип> (выражение);

Выражение: указатель или ссылка на класс

Тип: базовый или производный от типа выражения

Возвращаемое значение: переменную типа ‘тип’; при ошибке: 1) ноль (если ‘тип’ – указатель)

2) исключение bad_cast (если ‘тип’ – ссылка)

Пример:

```
// Повышающее преобразование (upcast)
class B { };
class C : public B { };
C* c = new C();
B* b = dynamic_cast<B*>(c);
```

dynamic_cast 2

Примеры: // понижающее преобразование (downcast)

1)
```
class A { };
class B : public A { };
A* a = new A();
B* b = dynamic_cast<B*>(a);
```

2)
```
class A {
    public: virtual void Test() {}
};
class B : public A {
    public: virtual void Test() {}
};
A* a = new A();
B* b = dynamic_cast<B*>(a);
```

dynamic_cast 3

Пример: // понижающее преобразование (downcast)

```
#include <typeinfo>
class B {
    public: virtual void Test();
};
class C : public B { };
void func(B& b) {
    try {
      C& c = dynamic_cast<C&>(b);
    } catch (bad_cast) {
    }
}
B* b = new B(); func(*b);  // исключение
C* c = new C(); func(*c);
```

dynamic_cast 4

Пример: // перекрёстное преобразование (crosscast)

```
class A {
    public: virtual void Test() {}
};
class B : public A {
};
class C : public A {
};
void func(C* c) {
    B* b = dynamic_cast<B*>(c);
}
B* b = new B();
func((C*)b);
```

dynamic_cast 5

Пример: // перекрёстное преобразование (crosscast)

```
class B {
    public: virtual void Test1() {}
};
class C {
    public: virtual void Test2() {}
};
class D : public B, public C {
};
void func(B* b) {
    C* c = dynamic_cast<C*>(b);
}
D* d = new D();
func(dynamic_cast<B*>(d));
```

static_cast 1

Синтаксис:

static_cast <тип> (выражение);

Выражение: целые, вещественные, перечисляемые типы; указатели и ссылки в одной иерархии

Возвращаемое значение: переменую типа ‘тип’

Пример:

```
float f = 10.0;
int i = static_cast<int>(f);
```

static_cast 2

Примеры:

```
class B { };
class C: public B { };
```

1) C c;

```
B* pb = static_cast<B*>(&c);
```

2) B b;

```
C& rc = static_cast<C&>(b);
```

reinterpret_cast

Синтаксис:

reinterpret_cast <тип> (выражение);

Выражение: любое

Тип: любой

Возвращаемое значение: переменная типа ‘тип’

Примеры:

```
char* p = reinterpret_cast<char*>(malloc(100));
long l = reinterpret_cast<long>(p);
```

typeid

Синтаксис:

1) typeid(тип);
2) typeid(выражение);

Примеры:

```
#include <typeinfo>
class B {
    public: virtual void Test(){ }
};
class C: public B { };
C* c = new C();
B* b = c;
if(typeid(c) == typeid(C*)) {
    C* cc = dynamic_cast<C*>(b);
}
printf("%s|%s|%s", typeid(int).name(), typeid(c).name(), typeid(*c).name());
```