Imię i nazwisko:..... **W** czas: 60 min

Przyjąć, że udostępniona jest przestrzeń nazw std

Zadanie 1 (5pkt)

Zmień implementację listy jednokierunkowej `List`, wykorzystaj sprytne wskaźniki.

```
class List {
  struct Node;
  typedef Node* PNode;
  struct Node {
    Node() { }
    ~Node() { }
    PNode next_;
  };
public:
  List() {}
  ~List() {
     while(head_) {
     PNode tmp = head_;
     head_ = head_->next_;
     delete tmp;
    }
  }
  void push_front() {
    PNode n = new Node();
    n->next_ = head_;
    head_ = n;
  }
private:
  PNode head_;
};
```

Zadanie 2 (3pkt)

Liczba liter Twojego nazwiska `LICZBA_LITER=` [ ]. Podaj napis generowany przez `zad2` [ ]

```
class E : public exception {
public:
  E(int i) : i_(i) {}
  int i_;
};
struct F {
  F(int i) : i_(i) { }
  virtual ~F() { cout << i_ << '␣';}
  int i_;
};
```

```
typedef unique_ptr<F> PF;
void f(PF&& pf) {
  if(pf->i_ > 3)
    f( PF(new F(pf->i_ - 3)));
  else
    throw E(pf->i_);
}
void f(const PF& pf) {
  if( pf->i_ > 1)
    f( PF(new F(pf->i_ / 2)));
}
```

```
void zad2() {
  try {
    PF pb(new F(LICZBA_LITER - 1));
    f(pb);
  } catch(E& e){
    cout << e.i_;
  }
  cout << endl;
}
```

Zadanie 3 (6pkt)

```
class BaseException : public exception {
public:
  virtual void accept(Visitor& v) = 0;
};
class Visitor {
public:
  virtual ~Visitor() {}
  virtual void visit(NetworkError&) = 0;
  virtual void visit(UserBreak&) = 0;
  virtual void visit(BadData&) = 0;
};
```

Popraw wydajność obsługi wyjątków w funkcji zad3().

```
class NetworkError : public BaseException {
public: virtual void accept(Visitor& v) { v.visit(*this); }
};
class UserBreak : public BaseException {
public: virtual void accept(Visitor& v) { v.visit(*this); }
};
class BadData : public BaseException {
public: virtual void accept(Visitor& v) { v.visit(*this); }
};
```

```
class Counters {
public:
  static Counters& getInstance() {
    static Counters counters;
    return counters;
  }
  int network_;
  int user_;
  int data_;
private:
  Counters() : network_(0), user_(0), data_(0) {}
  Counters(const Counters&) = delete;
  Counters& operator=(const Counters&) = delete;
};
```

```
void zad3() {
  try {

    //kod moze zglosic wyjatek typu BaseException

  } catch(NetworkError&) {
    ++(Counters::getInstance().network_);
  } catch(UserBreak&) {
    ++(Counters::getInstance().user_);
  } catch(BadData&) {
    ++(Counters::getInstance().data_);
  }
}
```

Zadanie 4 (3pkt)

Podaj napis generowany przez funkcję `zad4()`

```
struct B {
  int g() { return 1; }
};
struct D1 : virtual public B {
  int g() { return 2; }
};
struct D2 : virtual public B {
  virtual int g() { return 3; }
};
struct M : virtual public D1, virtual public D2 {
  virtual int g() { return 4; }
};
void zad4() {
  M m; D2* d2 = &m;
  cout << m.g() << d2->g() << static_cast<D1&>(m).g() << static_cast<B&>(m).g() << endl;
}
```

Zadanie 5 (6pkt)

Obiekty `Element` mogą tworzyć struktury drzewiaste. Dostarczyć możliwość tworzenia kopii głębokiej takich drzew.

```
class Element;
typedef shared_ptr<Element> PElement;
```

```
class Element {
public:
  Element() {}
  virtual ~Element(){}
  virtual void add(PElement) {}
};
```

```
class Leaf : public Element {
public:
  Leaf() : Element() { }
  ~Leaf() { }
};
```

```
class Node : public Element {
public:
  Node() : Element() { }
  ~Node() { }
  void add(PElement el) { ch_.push_back(el); }
private:
  vector<PElement> ch_;
};
```

Pytanie 1 (1pkt)

Dlaczego kod źródłowy powinien być czytelny ?

Pytanie 2 (1pkt)

Dlaczego używa się różnych języków programowania do tworzenia tej samej aplikacji ?

Uwagi do prowadzącego: