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

Przyjąć, że udostępniona jest przestrzeń nazw std

Zadanie 1 (5pkt)

Zmień implementację listy dwukierunkowej, użyj sprytnych wskaźników.

```
class List {
  struct Node {
    Node() : next_(nullptr), prev_(nullptr) {}
    Node* next_;
    Node* prev_;
  };
public:
    List() : head_(nullptr) {}
    ~List();
    void push_front();
private:
    Node* head_;
};
List::~List() {
    Node* n = head_;
    while( n != nullptr ) {
        Node* p = n;
        n = n->next_;
        delete p;
    }
}
void List::push_front() {
    Node* n = new Node();
    n->next_ = head_;
    if(head_) head_->prev_ = n;
    head_ = n;
}
```

Zadanie 2 (3pkt)

Podaj napis generowany przez `zad2` [ ], NAME to stała typu std::string zawierająca Twoje nazwisko zapisane wielkimi literami ASCII (zamiast 'Ą' jest 'A'), np. `WROZKA` dla Wróżka.

```
class E : public std::exception
{ };
class B {
public:
  B(int i) : i_(i) {
    cout << i_;
  }
  virtual ~B() { cout << i_; }
  int get() const { return i_; }
private:
  int i_;
};
class D1 : public B {
public:
  D1(int i) : B(i-1) {}
};
class D2 : public B {
public:
   D2(int i) : B(i+1) {}
};
class M : public D1, public D2 {
public:
  M(int i) : D1(i), D2(i) {}
  int get() const {
    return
    static_cast<const D1*>(this)->get();
  }
};
void f(int i, const M& m) {
  if(i <= 0) throw E();
  g(m.get());
}
void g(int i) {
  const int N = NAME.size() % 3 + 4;
  M m(N);
  f(m.get() - i, m);
}
void zad2() {
  try {
  } catch(E&){
  }
  cout << endl;
}
```

Zadanie 3 (4pkt)

Popraw wydajność obsługi wyjątków w funkcji zad3().

```
class Exception : public std::exception {
public:
  virtual void accept(Visitor& v) = 0;
};
class Visitor {
public:
  virtual ~Visitor() {}
  virtual void visit(MemoryError&) = 0;
  virtual void visit(AccessError&) = 0;
  virtual void visit(InputError&) = 0;
};
```

```
class MemoryError : public Exception {
public:
  virtual void accept(Visitor& v) { v.visit(*this); }
};
class AccessError : public Exception {
public:
  virtual void accept(Visitor& v) { v.visit(*this); }
};
class InputError : public Exception {
public:
  virtual void accept(Visitor& v) { v.visit(*this); }
};
```

```
void zad3() {
  try {
    //kod moze zglosic wyjatek z hierarchii Exception
  } catch(MemoryError&) {
     cerr << "Memory␣Error" << endl;
  } catch(AccessError&) {
     cerr << "Access␣Error" << endl;
  } catch(InputError&) {
     cerr << "Input␣Error" << endl;
  }
```

Zadanie 4 (6pkt)

Fabryka tworzy obiekty reprezentujące datę oraz czas; proszę zapewnić, że będą tworzone zgodne typy za pomocą tego samego obiektu fabryki.

```
Factory* f; //tutaj ma byc inicjacja
f->date(2018,04,10);
f->time(20,0,0);
```

```
class Time {
public:
  virtual ~Time() {}
};
using PTime = std::shared_ptr<Time>;
class Date {
public:
  virtual ~Date() {}
};
using PDate = std::shared_ptr<Date>;
```

```
class BoostTime : public Time { public: BoostTime(int h, int m, int s); };
class ChronoTime : public Time { public: ChronoTime(int h, int m, int s); };
class BoostDate : public Date { public: BoostDate(int y, int m, int d); };
class ChronoDate : public Date { public: ChronoDate(int y, int m, int d); };

class Factory {
public:
  virtual PTime time(int h, int m, int s) = 0;
  virtual PDate date(int y, int m, int d) = 0;
};
```

Zadanie 5 (6pkt)

Wywołanie metody `close` dla okna nadrzędnego (`MainView`) powinno uruchomić metody `close` dla wszystkich otwartych okien podrzędnych (obiekty `ChildViewA`, `ChildViewB`). Zakładamy, że okno jest otwarte, gdy wywołano metodę `open`. Obok pokazano test. Proszę uzupełnić implementację.

```
MainView m;
ChildViewA ch1(&m);
ch1.open();
ChildViewB ch2(&m);
ch2.open();
ChildViewA ch3(&m);
m.close();
```

```
class MainView {
public:
   void close() { }
};


class ChildViewA : public Observer {
public:
   ChildViewA(MainView* m) : m_(m) {}
   void open() { }
   void close() { }
private:
   MainView* m_;
};

class ChildViewB : public Observer {
public:
   ChildViewB(MainView* m) : m_(m) {}
   void open() { }
   void close() { }
private:
   MainView* m_;
};
```

Pytanie 1 (1pkt)

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

Uwagi do prowadzącego: