LAMPIRAN A

Listing Program

Form1.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
using System.IO;
using System.Threading;
using System.Collections;

namespace GAScrabble
{

    public delegate void SetScrabbleDisplayDelegate(string
displayText);

    public partial class Form1 : Form
    {
        bool _stop = false;

        public Form1()
        {
            InitializeComponent();
        }

        private string _txtJawaban = "";

        private void AlgoritmaGenetik()
        {
            PopulasiScrabble population = new PopulasiScrabble();

            for (int i = 0; i < 100000; i++)
            {
                population.NextGeneration();

                if (i % 50 == 0)
                {
                    population.GetScoreGenomeTerbesar();
                    _txtJawaban =
population.GetTopGenome().ToString();
                    this.BeginInvoke(new
SetScrabbleDisplayDelegate(SetScrabbleDisplay), new object[] {
_txtJawaban });
                }

                if (_stop == true)
                {
                    _stop = false;
                    break;
                }
            }
        }
```

```
        private void SetScrabbleDisplay(string displayText)
        {
            lblAnswer.Text = _txtJawaban;

                if (listBox1.Items.Contains(_txtJawaban))
                {
                    listBox1.Items.Clear();
                }

             listBox1.Items.AddRange(new string[] { _txtJawaban
});
        }


        private void btnContinuous_Click(object sender, EventArgs
e)
        {
            ScrabbleGenome.InisialHurufScrabble = TextBox.Text;
            _stop = false;

            Thread oThread = new Thread(new
ThreadStart(AlgoritmaGenetik));
            oThread.Start();
        }

        private void Form1_Paint(object sender, PaintEventArgs e)
        {
        }

        private void btnStop_Click(object sender, EventArgs e)
        {
            _stop = true;
        }

        private void Form1_FormClosing(object sender,
FormClosingEventArgs e)
        {
            _stop = true;
        }

        private void Form1_Load(object sender, EventArgs e)
        {
        }
    }
}
```

Population.cs

```
using System;
using System.Collections;

namespace GAScrabble
{

      public class Populasi
      {

            public const int kPanjang = 8;
            protected const int kCrossover = kPanjang/2;
            protected const int kInisialPopulasi = 1000;
            protected const int kBatasPopulasi = 1000;
            protected const int kMin = 1;
            protected const int kMax = 1000;
            protected const float kFrekuensiMutasi = 0.10f;
            protected const float kKecocokanKematian = 0.00f;
            protected const float kKecocokanReproduksi = 0.0f;

            protected ArrayList Genomes = new ArrayList();
            protected ArrayList ReproduksiGenome  = new
ArrayList();
            protected ArrayList HasilGenome = new ArrayList();
            protected ArrayList FamilyGenome = new ArrayList();

            protected int JumlahPopulasi = kInisialPopulasi;
            protected int Generasi = 1;
            protected bool Best2 = true;

            public Genome GetScoreGenomeTerbesar()
            {
                  Genomes.Sort();
                  return (Genome)Genomes[0];
            }

            public virtual void CatatNextGeneration()
            {
                  Console.WriteLine("Generation {0}\n", Generasi);
                  if (Generasi % 1 == 0)
                  {
                        Genomes.Sort();
                    int count = 0;
                    foreach (Genome g in Genomes)
                    {
                        count++;
                        if (count == 10)
                            break;
                    }
                      }
            }

        public virtual void TopGenome()
        {
            if (Generasi % 1 == 0) //hanya dicetak setiap 100
generasi
```

```
            {
                Genomes.Sort();
                Genome g = Genomes[0] as Genome;
            }
        }

        public virtual Genome GetTopGenome()
        {
            Genomes.Sort();
            Genome g = Genomes[0] as Genome;
            return g;
        }
    }
}
```

ScrabblePopulation.cs

```
using System;
using System.Collections;

namespace GAScrabble
{

      public class PopulasiScrabble : Populasi
      {
            public PopulasiScrabble()
            {
                  Genomes.Clear();
                  for  (int i = 0; i < kInisialPopulasi; i++)
                  {
                ScrabbleGenome aGenome = new
GAScrabble.ScrabbleGenome(ScrabbleGenome.PanjangKata, 1,
ScrabbleGenome.PanjangKata);
                        aGenome.SetCrossoverPoint(kCrossover);
                        aGenome.KalkulasiKecocokan();
                        Genomes.Add(aGenome);
                  }
            }

            public void NextGeneration()
            {
                  Generasi++;

                  for  (int i = 0; i < Genomes.Count; i++)
                  {
                        if
(((Genome)Genomes[i]).YangMati(kKecocokanKematian))
                        {
                              Genomes.RemoveAt(i);
                              i--;
                        }
                  }

                  ReproduksiGenome.Clear();
                  HasilGenome.Clear();
                  for  (int i = 0; i < Genomes.Count; i++)
                  {
                        if
(((Genome)Genomes[i]).YangReproduksi(kKecocokanReproduksi))
                        {
                              ReproduksiGenome.Add(Genomes[i]);

                        }
                  }

                  DoCrossover(ReproduksiGenome);
                  Genomes = (ArrayList)HasilGenome.Clone();
                  for  (int i = 0; i < Genomes.Count; i++)
                  {
                ((ScrabbleGenome)Genomes[i]).KalkulasiKecocokan();
                  }
```

```
                  if (Genomes.Count > kBatasPopulasi)
                  Genomes.RemoveRange(kBatasPopulasi,
Genomes.Count - kBatasPopulasi);
                  JumlahPopulasi = Genomes.Count;
            }

            public void KalkulasiKecocokanUntukSmua(ArrayList
gens)
            {
                  foreach(ScrabbleGenome lg in gens)
                  {
                        lg.KalkulasiKecocokan();
                  }
            }

            public void DoCrossover(ArrayList gen)
            {
                  ArrayList GenInduk = new ArrayList();
                  ArrayList GenAyah = new ArrayList();

                  for (int i = 0; i < gen.Count; i++)
                  {
                        if (ScrabbleGenome.Bibit.Next(100) % 2 >
0)
                        {
                              GenInduk.Add(gen[i]);
                        }
                        else
                        {
                              GenAyah.Add(gen[i]);
                        }
                  }

                  if (GenInduk.Count > GenAyah.Count)
                  {
                        while (GenInduk.Count > GenAyah.Count)
                        {
                              GenAyah.Add(GenInduk[GenInduk.Count
- 1]);
                              GenInduk.RemoveAt(GenInduk.Count -
1);
                        }

                        if (GenAyah.Count > GenInduk.Count)
                        {
                              GenAyah.RemoveAt(GenAyah.Count - 1);
// make sure they are equal
                        }
                  }
                  else
                  {
                        while (GenAyah.Count > GenInduk.Count)
                        {
                              GenInduk.Add(GenAyah[GenAyah.Count -
1]);
                              GenAyah.RemoveAt(GenAyah.Count - 1);
                        }
```

```
                        if (GenInduk.Count > GenAyah.Count)
                        {
                              GenInduk.RemoveAt(GenInduk.Count -
1);
                        }
                  }

                  for (int i = 0; i < GenAyah.Count; i ++)
                  {
                ScrabbleGenome GenAnak1 =
(ScrabbleGenome)((ScrabbleGenome)GenAyah[i]).Clone();
                ScrabbleGenome GenAnak2 =
(ScrabbleGenome)((ScrabbleGenome)GenInduk[i]).Clone();

                        FamilyGenome.Clear();
                        FamilyGenome.Add(GenAyah[i]);
                        FamilyGenome.Add(GenInduk[i]);
                        FamilyGenome.Add(GenAnak1);
                        FamilyGenome.Add(GenAnak2);
                        KalkulasiKecocokanUntukSmua(FamilyGenome);
                        FamilyGenome.Sort();

                        if (Best2 == true)
                        {
                              HasilGenome.Add(FamilyGenome[0]);

                              HasilGenome.Add(FamilyGenome[1]);
                        }
                        else
                        {
                              HasilGenome.Add(GenAnak1);
                              HasilGenome.Add(GenAnak2);
                        }
                  }
            }
      }
}
```

Genome.cs

```
using System;
using System.Collections;

namespace GAScrabble
{

      public abstract class Genome : IComparable
      {
            public long Panjang;
            public int  CrossoverPoint;
            public int  IndexMutasi;
            public float JumlahKecocokan = 0.0f;

        abstract public object HasilkanNilaiGen(object min, object
max);
        abstract public object HasilkanGen();
        abstract public void SetCrossoverPoint(int
crossoverPoint);
            abstract public bool  YangReproduksi(float fitness);
            abstract public bool  YangMati(float fitness);
            abstract public void    CopyInfoGen(Genome g);

            abstract public int CompareTo(object a);
      }
}
```

ListGenome.cs

```
using System;
using System.Collections;
using System.Collections.Generic;

namespace GAScrabble
{
      public class DaftarGenome : Genome
      {
            protected ArrayList TheArray = new ArrayList();
            public static Random Bibit = new
Random((int)DateTime.Now.Ticks);
            protected int TheMin = 0;
            protected int TheMax = 100;

        public override object HasilkanGen()
        {
            return this;
        }

        public override int CompareTo(object a)
            {
                  DaftarGenome Gen1 = this;
                  DaftarGenome Gen2 = (DaftarGenome)a;
                  return Math.Sign(Gen2.JumlahKecocokan  -
Gen1.JumlahKecocokan);
            }

            public override void SetCrossoverPoint(int
crossoverPoint)
            {
                  CrossoverPoint = crossoverPoint;
            }

            public DaftarGenome()
            {
            }

            public DaftarGenome(long length, object min, object
max)
            {
                  Panjang = length;
                  TheMin = (int)min;
                  TheMax = (int)max;
                  for (int i = 0; i < Panjang; i++)
                  {
                     int nextValue = (int)HasilkanNilaiGen(min,
max);
                     TheArray.Add(nextValue);
                  }
            }

            public override bool YangMati(float kecocokan)
            {
                  if (JumlahKecocokan <= (int)(kecocokan *
100.0f))
```

```
                  {
                        return true;
                  }

                  return false;
            }

            public override bool YangReproduksi(float fitness)
            {
                  if (DaftarGenome.Bibit.Next(100) >=
(int)(fitness * 100.0f))
                  {
                        return true;
                  }

                  return false;
            }

            public override object HasilkanNilaiGen(object min,
object max)
            {
                  return Bibit.Next((int)min, (int)max);
            }

            public override void CopyInfoGen(Genome dest)
            {
                  DaftarGenome theGen = (DaftarGenome)dest;
                  theGen.Panjang = Panjang;
                  theGen.TheMin = TheMin;
                  theGen.TheMax = TheMax;
            }
      }
}
```

ScrabbleGenome.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using NetSpell.SpellChecker;

namespace GAScrabble
{
      public class ScrabbleGenome : DaftarGenome, ICloneable
      {
        protected List<char> _Penyanggascrabble = new
List<char>();

        static protected char[] _inisialHurufScrabble = new char[]
{ 'I', 'C', 'O', 'N', 'S'};


        public static int PanjangKata
        {
            get
            {
                return _inisialHurufScrabble.Length;
            }
        }

       public static  string InisialHurufScrabble
        {
            set
            {
                _inisialHurufScrabble = new char[value.Length];
                int i = 0;
                foreach (char c in value.ToCharArray())
                {
                    _inisialHurufScrabble[i] = c;
                    i++;
                }
            }
        }

        private void PopulasiPenyanggaScrabble()
        {
            _Penyanggascrabble.Clear();
            _Penyanggascrabble.AddRange(_inisialHurufScrabble);
        }

        public override object HasilkanGen()
        {
            int length = PanjangKata;
            PopulasiPenyanggaScrabble();
            TheArray.Clear();
            for (int i = 0; i < length; i++)
            {
                int nextCharIndex =
Bibit.Next(_Penyanggascrabble.Count - 1);
                TheArray.Add(_Penyanggascrabble[nextCharIndex]);
                _Penyanggascrabble.RemoveAt(nextCharIndex);
```

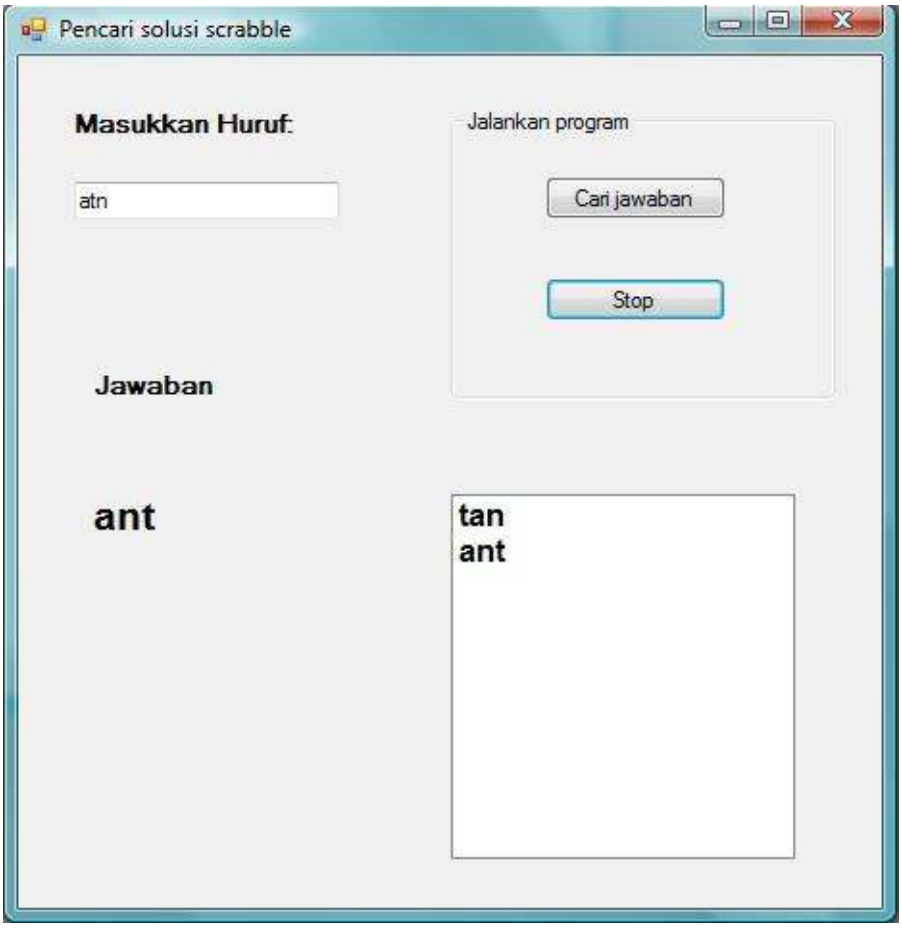
```
            }

            return this;
        }

            public ScrabbleGenome()
            {
            }

            public ScrabbleGenome(long length, object min, object
max) : base(length, min, max)
            {
            HasilkanGen();
            }

        static private Spelling _speller = new Spelling();

            private float KalkulasiDariGAScrabble()
            {
                  float fScoreKecocokan = 0.0f;
            string word = ToString();

            if (_speller.TestWord(word))
            {
                fScoreKecocokan = TheArray.Count;
            }
            else
            {
                fScoreKecocokan = .01f;
            }

                  return fScoreKecocokan;
            }

            public  float KalkulasiKecocokan()
            {
                JumlahKecocokan = KalkulasiDariGAScrabble();

                  return JumlahKecocokan;
            }

            public override string ToString()
            {
                  string strHasil = "";

            foreach (char nilai in TheArray)
            {
                strHasil += nilai;
            }

                  return strHasil;
            }

            public  void CopyInfoGen(Genome dest)
            {
                  ScrabbleGenome theGene = (ScrabbleGenome)dest;
                  theGene.Panjang = Panjang;
                  theGene.TheMin = TheMin;
```

```
                theGene.TheMax = TheMax;
            }

        public object Clone()
        {
            ScrabbleGenome g = new ScrabbleGenome();
            g.TheArray = (ArrayList)TheArray.Clone();
            g.Panjang = Panjang;

            return g;
        }
    }
}
```

LAMPIRAN B

DOKUMENTASI

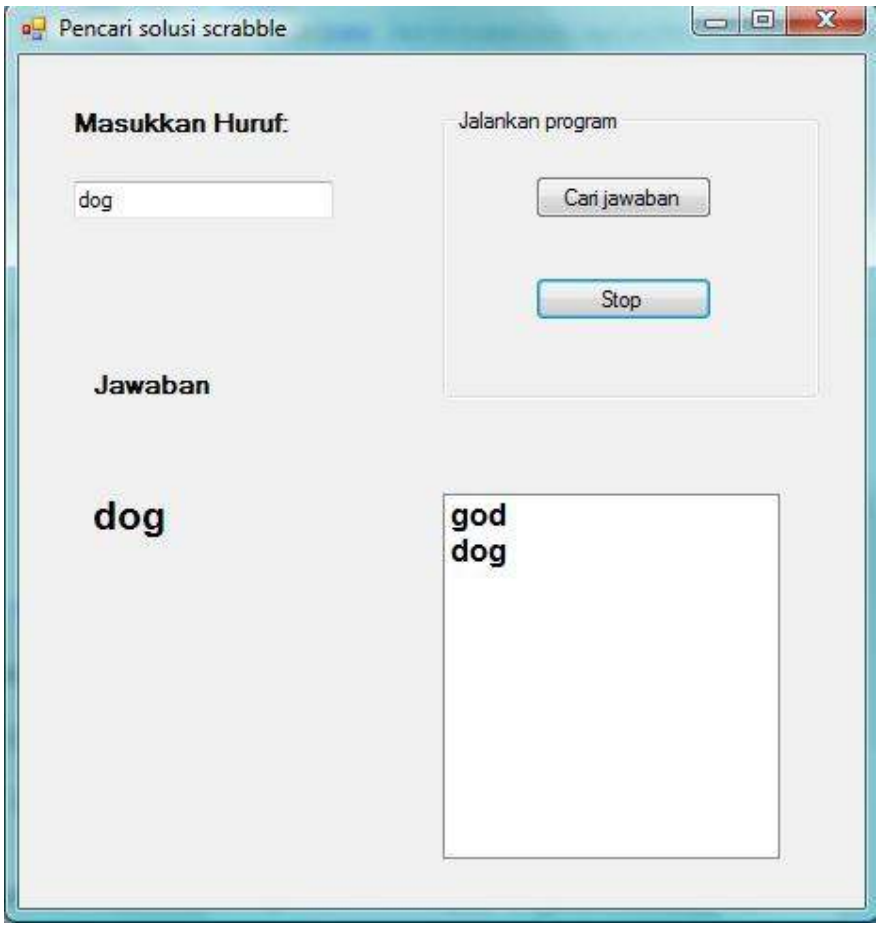
Percobaan Pada Tiga Huruf




Pencari solusi scrabble
Masukkan Huruf:
eap
Jalankan program
Cari jawaban
Stop
Jawaban
ape
pea
ape


Pencari solusi scrabble
Masukkan Huruf:
ahs
Jalankan program
Cari jawaban
Stop
Jawaban
ash
has
ash


Pencari solusi scrabble
Masukkan Huruf:
ate
Jalankan program
Cari jawaban
Stop
Jawaban
eat
ate
eat


Pencari solusi scrabble
Masukkan Huruf:
dog
Jalankan program
Cari jawaban
Stop
Jawaban
dog
god
dog


Pencari solusi scrabble
Masukkan Huruf:
bna
Jalankan program
Cari jawaban
Stop
Jawaban
nab
ban
nab


Pencari solusi scrabble
Masukkan Huruf:
atr
Jalankan program
Cari jawaban
Stop
Jawaban
rat
tar
art
rat

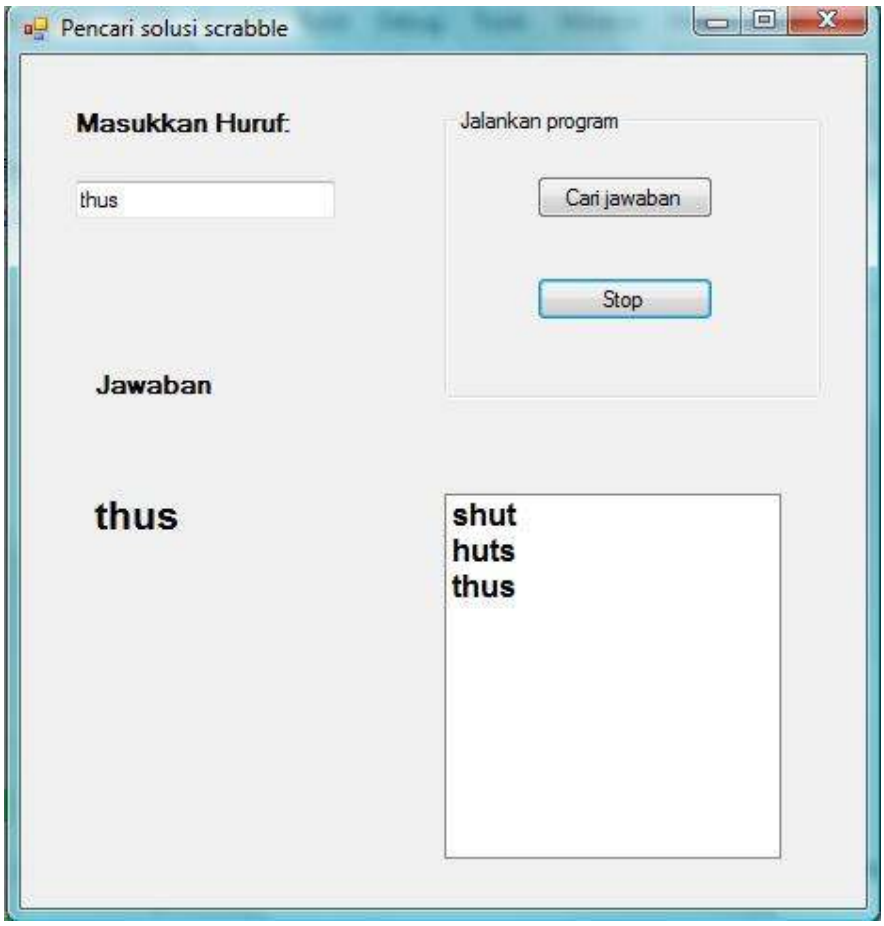
Percobaan Pada Empat Huruf

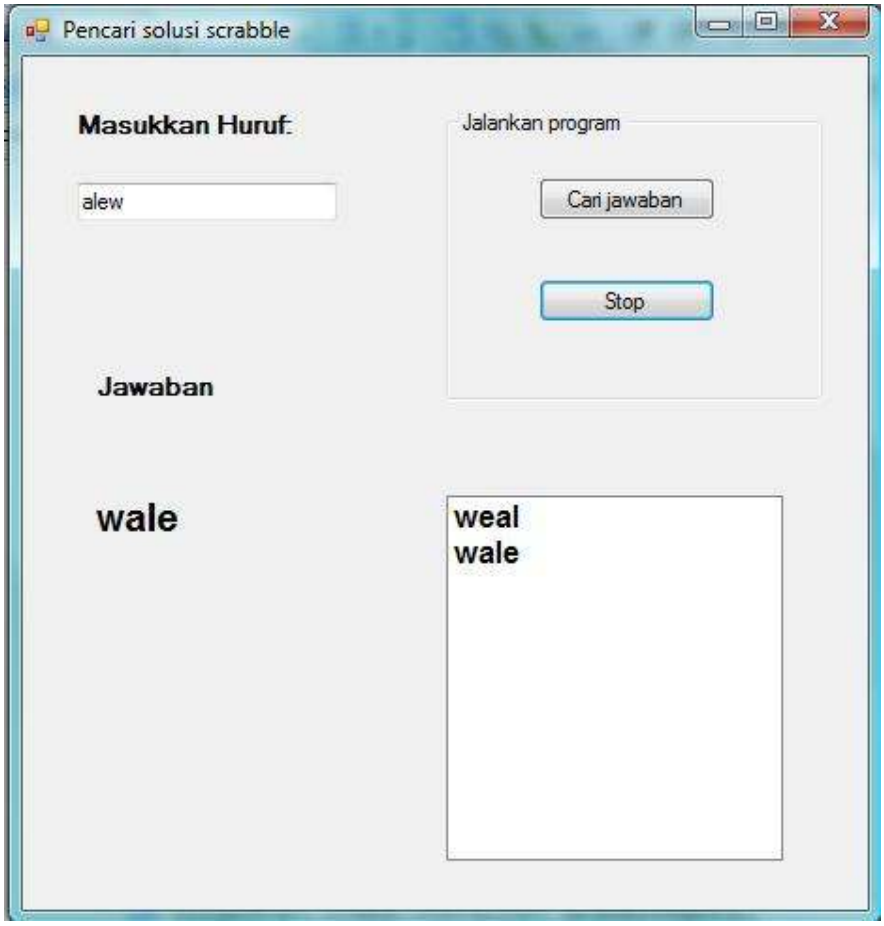

Pencari solusi scrabble
Masukkan Huruf:
amef
Jalankan program
Cari jawaban
Stop
Jawaban
fame
fame


Pencari solusi scrabble
Masukkan Huruf:
earh
Jalankan program
Cari jawaban
Stop
Jawaban
hare
hear
rhea
hare

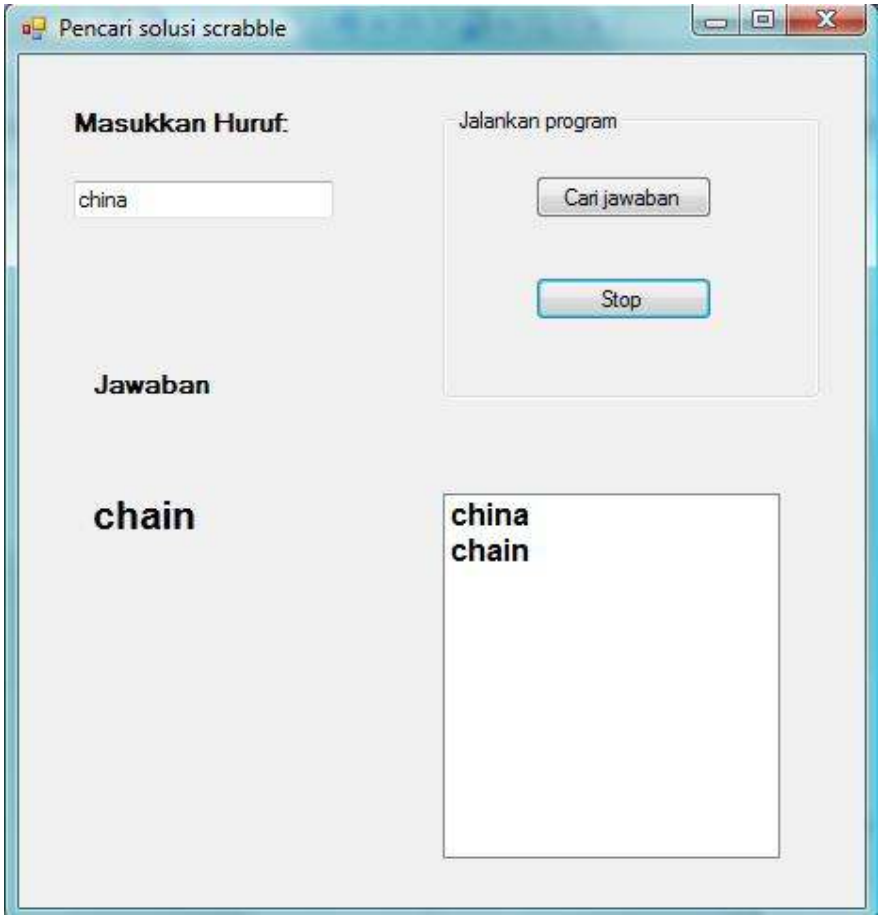

Pencari solusi scrabble
Masukkan Huruf:
eamt
Jalankan program
Cari jawaban
Stop
Jawaban
mate
meta
meat
team
tame
mate

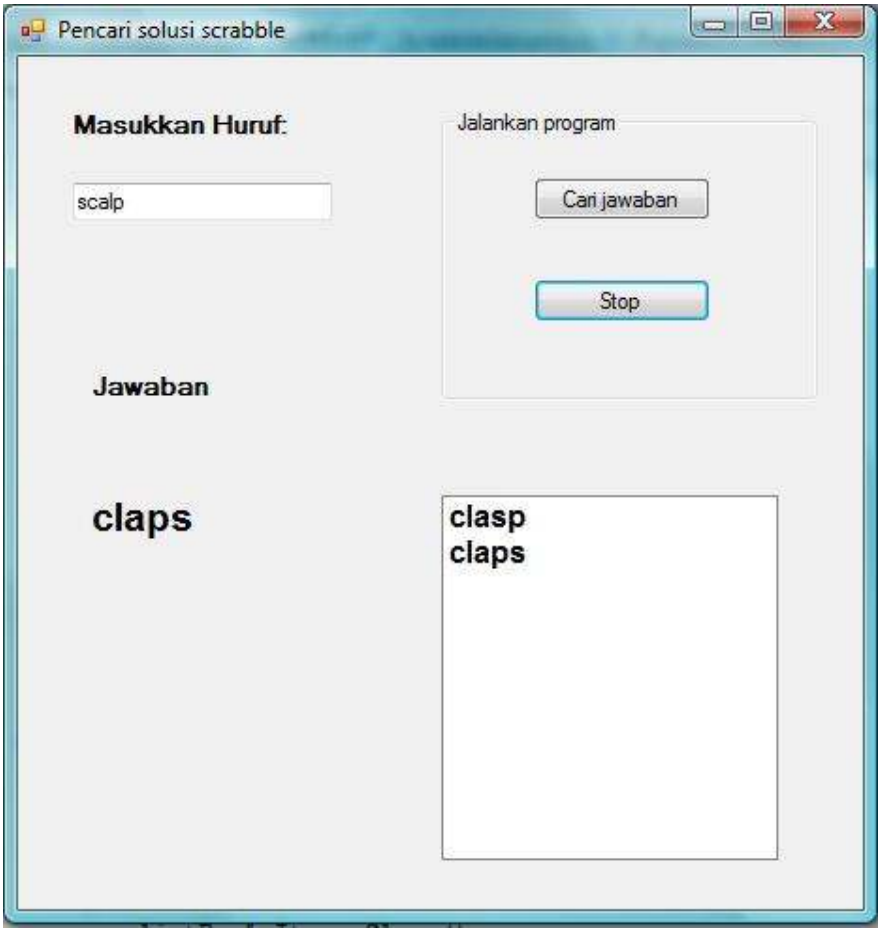

Pencari solusi scrabble
Masukkan Huruf:
eyrp
Jalankan program
Cari jawaban
Stop
Jawaban
prey
pyre
prey


Pencari solusi scrabble
Masukkan Huruf:
thus
Jalankan program
Cari jawaban
Stop
Jawaban
thus
shut
huts
thus


Pencari solusi scrabble
Masukkan Huruf:
alew
Jalankan program
Cari jawaban
Stop
Jawaban
wale
weal
wale

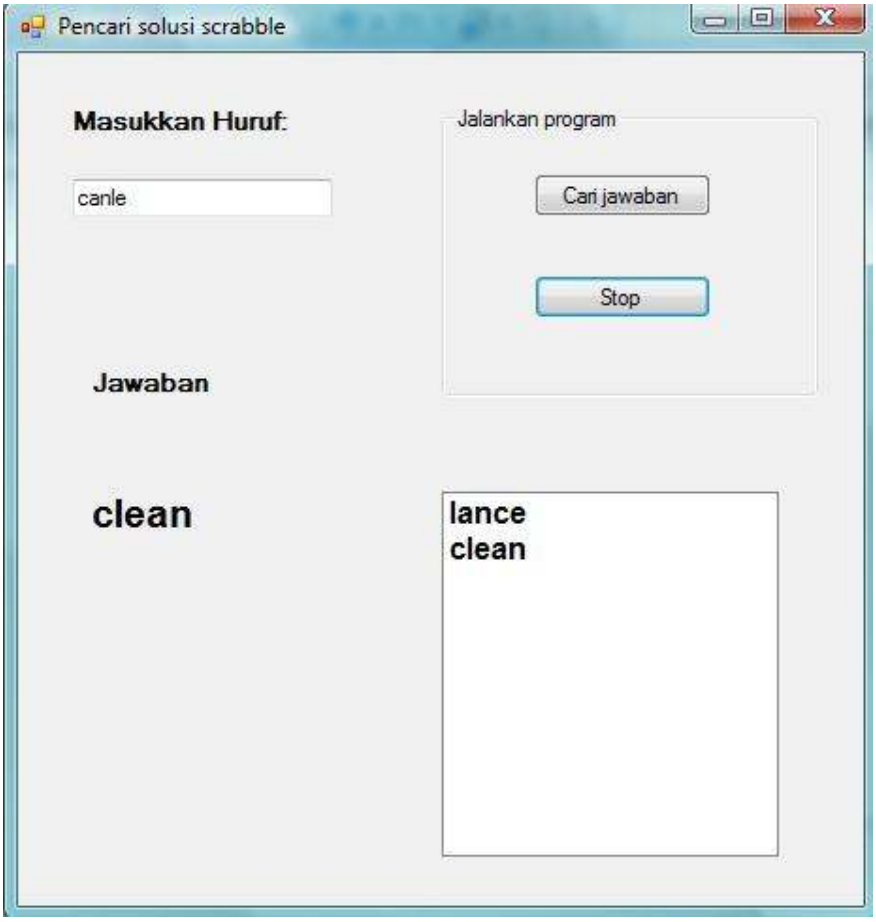
Percobaan Pada Lima Huruf


Pencari solusi scrabble
Masukkan Huruf:
china
Jalankan program
Cari jawaban
Stop
Jawaban
chain
china
chain


Pencari solusi scrabble
Masukkan Huruf:
scalp
Jalankan program
Cari jawaban
Stop
Jawaban
claps
clasp
claps


Pencari solusi scrabble
Masukkan Huruf:
frdei
Jalankan program
Cari jawaban
Stop
Jawaban
fried
fried


Pencari solusi scrabble
Masukkan Huruf:
dexin
Jalankan program
Cari jawaban
Stop
Jawaban
index
index


Pencari solusi scrabble
Masukkan Huruf:
canle
Jalankan program
Cari jawaban
Stop
Jawaban
clean
lance
clean

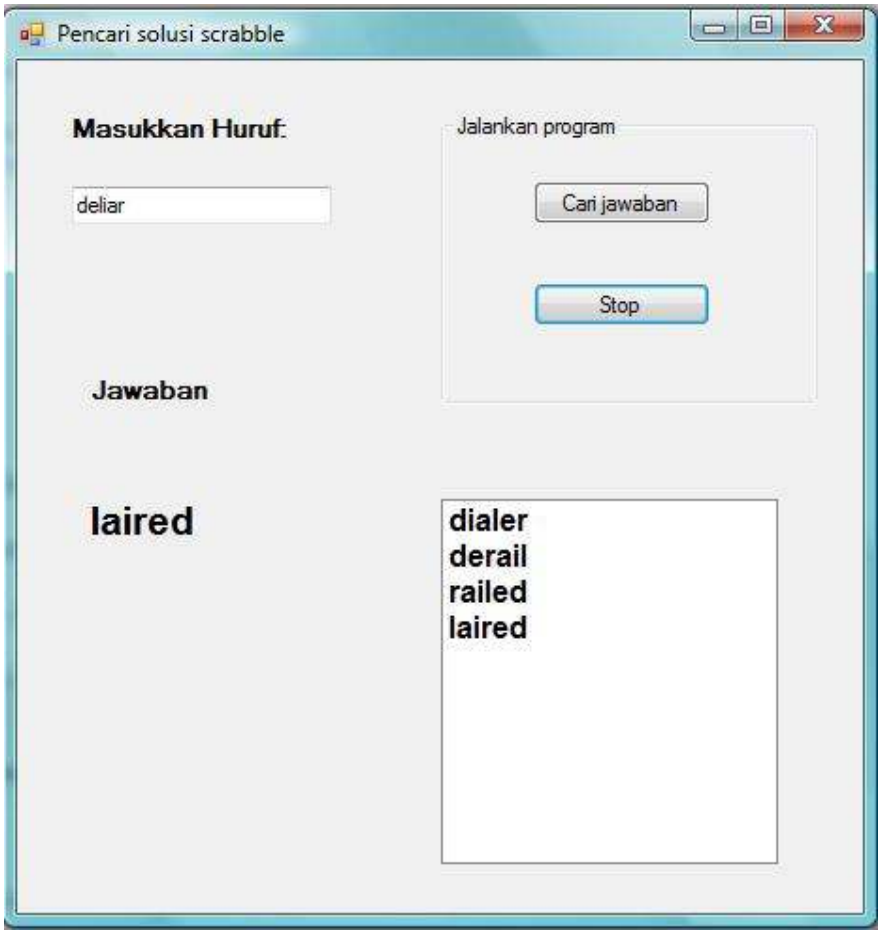

Pencari solusi scrabble
Masukkan Huruf:
depta
Jalankan program
Cari jawaban
Stop
Jawaban
pated
adept
taped
pated

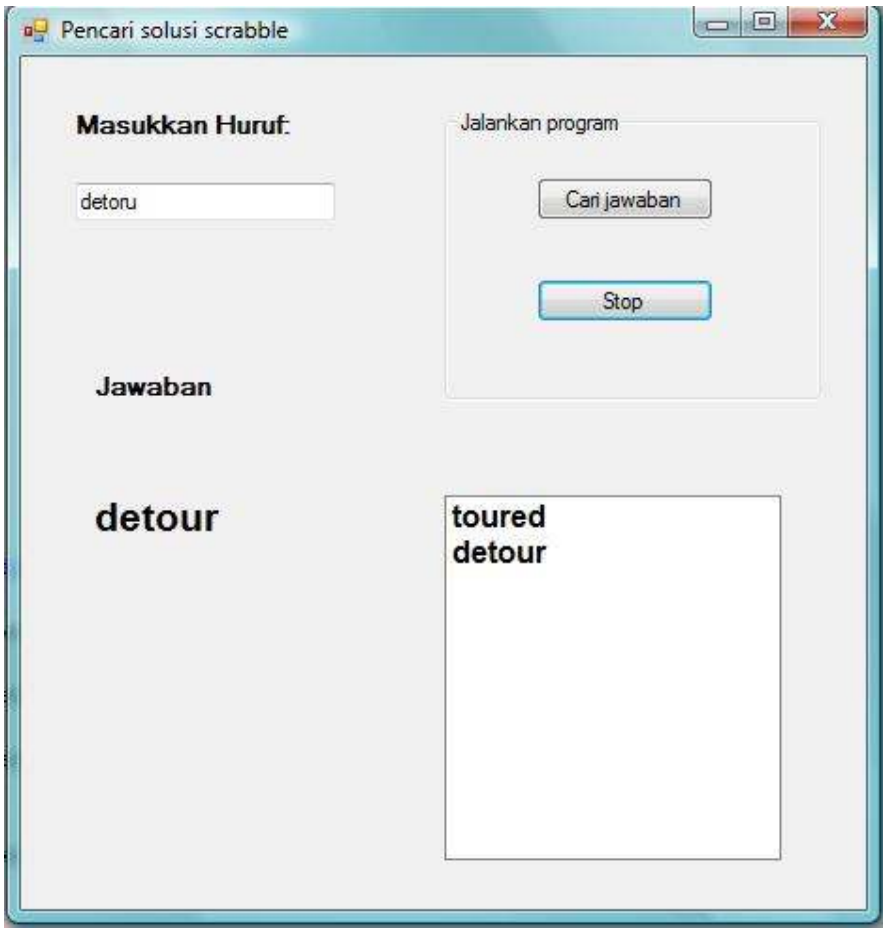
Percobaan Pada Enam Huruf


Pencari solusi scrabble
Masukkan Huruf:
boderr
Jalankan program
Cari jawaban
Stop
Jawaban
border
border


Pencari solusi scrabble
Masukkan Huruf:
sucaal
Jalankan program
Cari jawaban
Stop
Jawaban
casual
causal
casual

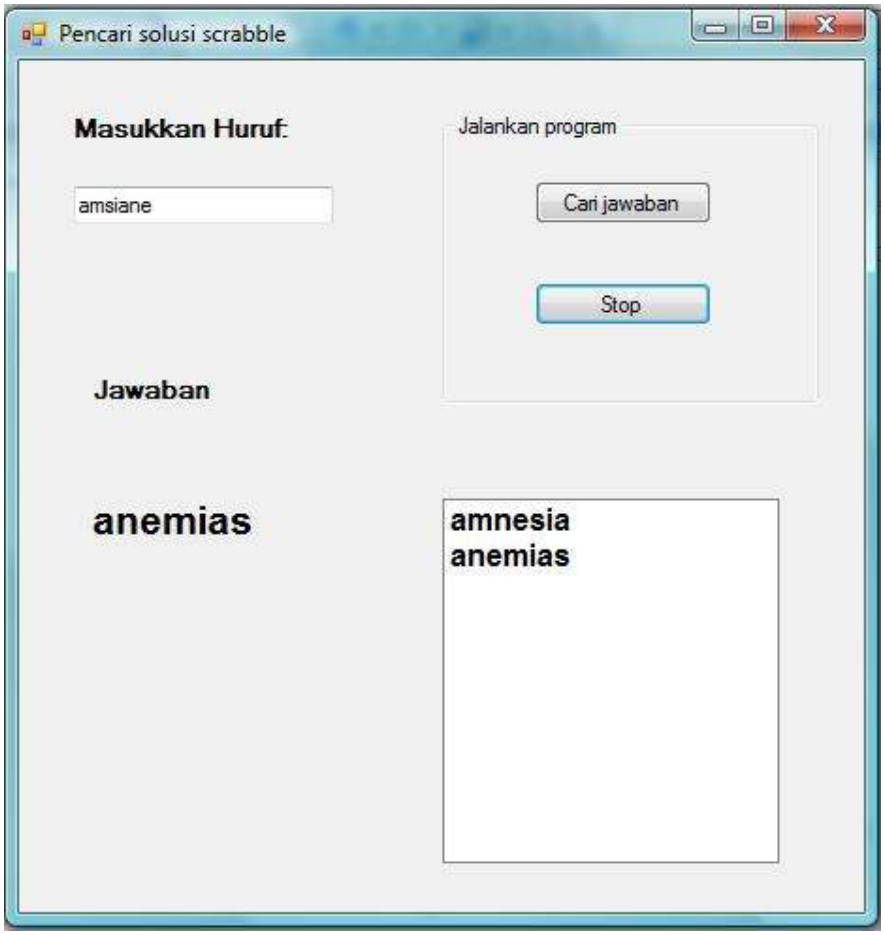

Pencari solusi scrabble
Masukkan Huruf:
paetdr
Jalankan program
Cari jawaban
Stop
Jawaban
depart
prated
parted
petard
depart


Pencari solusi scrabble
Masukkan Huruf:
deliar
Jalankan program
Cari jawaban
Stop
Jawaban
laired
dialer
derail
railed
laired


Pencari solusi scrabble
Masukkan Huruf:
detoru
Jalankan program
Cari jawaban
Stop
Jawaban
detour
toured
detour


Pencari solusi scrabble
Masukkan Huruf:
rusebi
Jalankan program
Cari jawaban
Stop
Jawaban
rubies
busier
buries
bruise
rubies

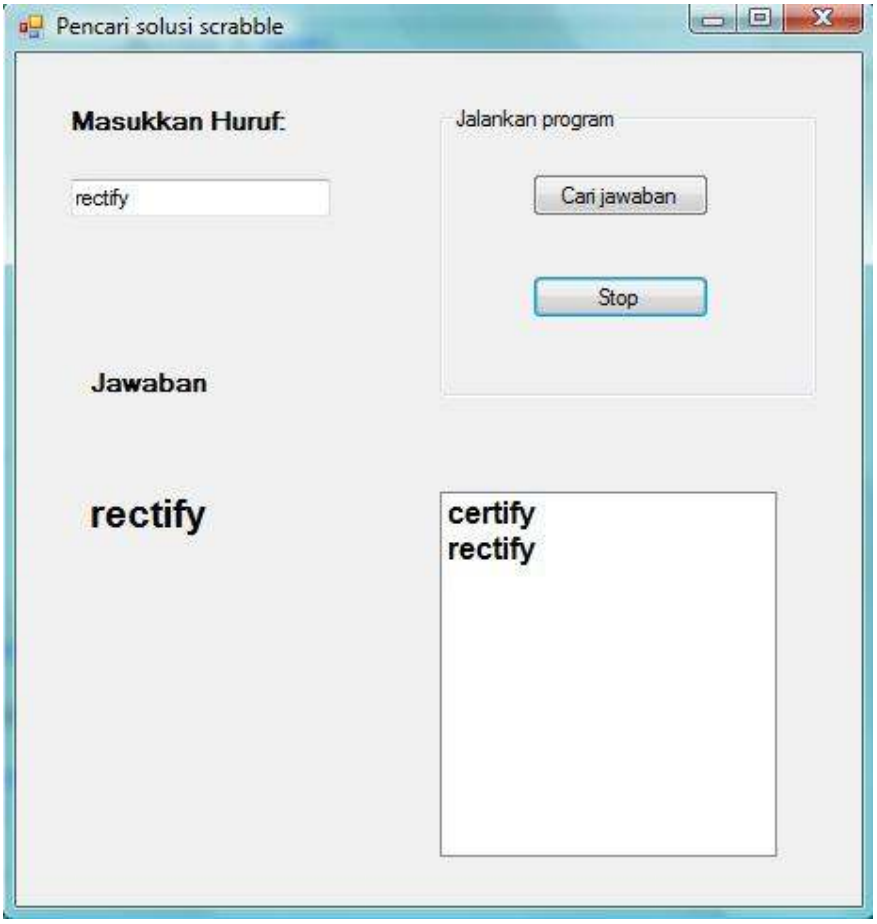
Percobaan Pada Tujuh Huruf

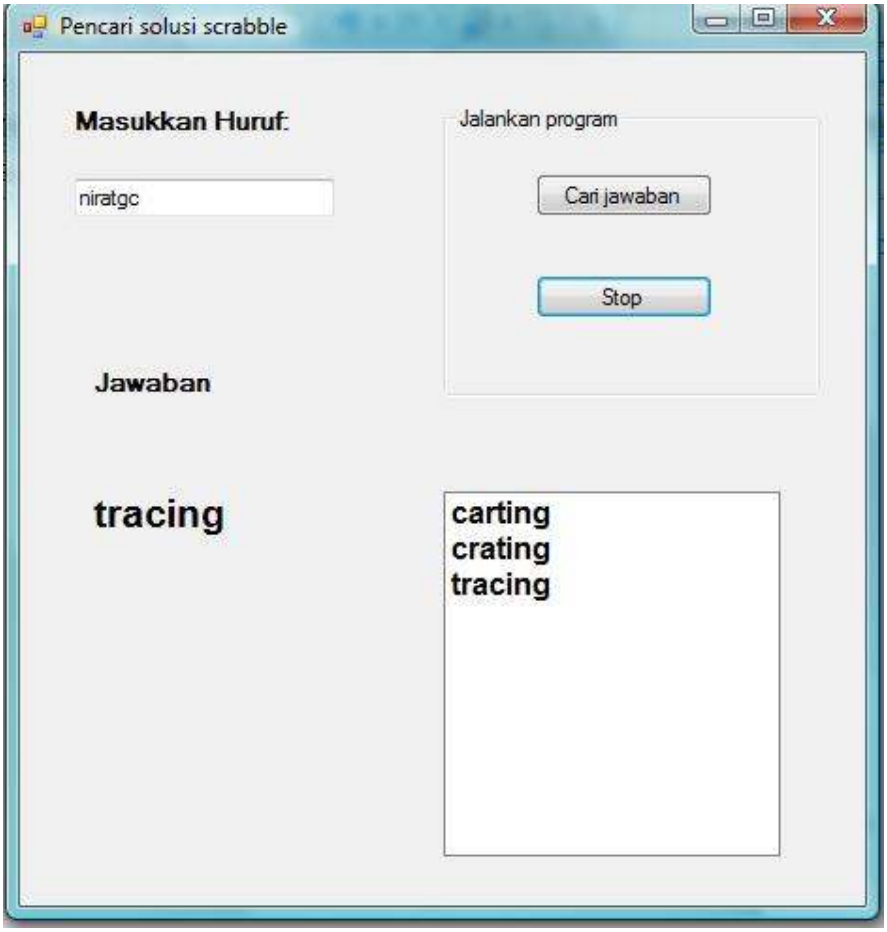



Pencari solusi scrabble
Masukkan Huruf:
ciotnua
Jalankan program
Cari jawaban
Stop
Jawaban
caution
auction
caution


Pencari solusi scrabble
Masukkan Huruf:
singcha
Jalankan program
Cari jawaban
Stop
Jawaban
cashing
chasing
cashing


Pencari solusi scrabble
Masukkan Huruf:
negelar
Jalankan program
Cari jawaban
Stop
Jawaban
general
enlarge
general


Pencari solusi scrabble
Masukkan Huruf:
rectify
Jalankan program
Cari jawaban
Stop
Jawaban
rectify
certify
rectify

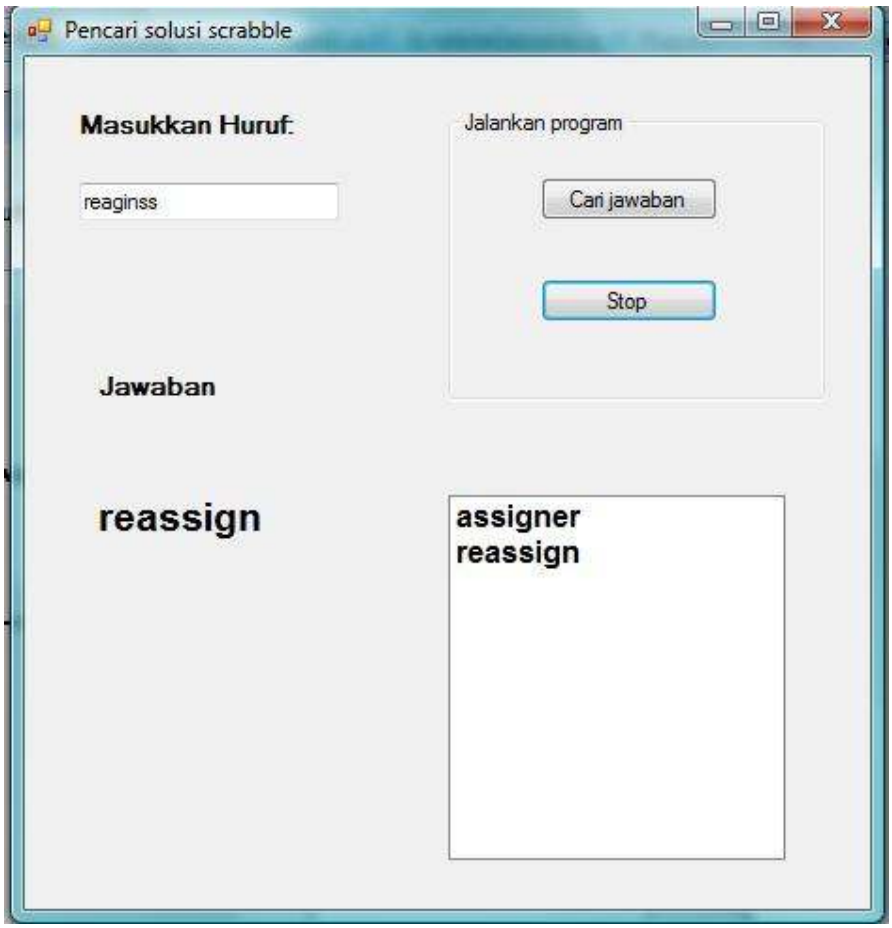

Pencari solusi scrabble
Masukkan Huruf:
niratgc
Jalankan program
Cari jawaban
Stop
Jawaban
tracing
carting
crating
tracing


Pencari solusi scrabble
Masukkan Huruf:
rturenu
Jalankan program
Cari jawaban
Stop
Jawaban
nurture
nurture

Percobaan Pada Delapan Huruf


Pencari solusi scrabble
Masukkan Huruf:
sncation
Jalankan program
Cari jawaban
Stop
Jawaban
contains
canonist
sanction
contains


Pencari solusi scrabble
Masukkan Huruf:
gasinpir
Jalankan program
Cari jawaban
Stop
Jawaban
pairings
praising
aspiring
pairings


Pencari solusi scrabble
Masukkan Huruf:
reaginss
Jalankan program
Cari jawaban
Stop
Jawaban
reassign
assigner
reassign


Pencari solusi scrabble
Masukkan Huruf:
rraonagt
Jalankan program
Cari jawaban
Stop
Jawaban
tarragon
arrogant
tarragon


Pencari solusi scrabble
Masukkan Huruf:
reatpord
Jalankan program
Cari jawaban
Stop
Jawaban
teardrop
predator
parroted
teardrop


Pencari solusi scrabble
Masukkan Huruf:
laerting
Jalankan program
Cari jawaban
Stop
Jawaban
altering
alerting
triangle
integral
relating
altering

LAMPIRAN C

KAMUS REFERENSI KATA-KATA YANG DIGUNAKAN

aboard
abroad
adept
alert
alter
amnesia
anemia
ant
ape
arrogant
Art
art
ash
Ash
aspire
assigner
astatine
ate
auction
Ban
ban
beast
beat
beta
border
bruise
bury
busy
canonist
cart
Cart
cash
Cash
casual
causal
caution
certify
chain
China
china
clap
chasing
clasp
clean
contain
crate
depart

derail
detour
dialer
dog
doom
eat
enlarge
fame
fried
general
God
god
hare
has
hear
hut
index
integral
lair
Lance
lance
mate
meat
meta
Meta
mood
nab
nurture
pair
parrot
part
Pat
pat
pea
petard
praising
prate
predator
prey
pyre
rail
rat
reassign
rectify
relate
Rhea
rhea

Ruby
ruby
sanction
sanitate
shut
tame
Tan
tan
tape
taped
tar
tarragon
team
teardrop
thus
toured
tracing
triangle
wale
weal