Portfolio
Finding Relevant Functions and their Usages

Collin McMillan\textsuperscript{1}
Mark Grechanik\textsuperscript{2}
Denys Poshivanyk\textsuperscript{1}
Qing Xie\textsuperscript{2}
Chen Fu\textsuperscript{2}

\textsuperscript{1}College of William & Mary
\textsuperscript{2}Accenture Technology Labs
Virtues of Programmers:

- Laziness
- Impatience
- Hubris

Larry Wall,
Inventor of Perl
Example Programming Task

Write a utility for dithering mip map images that are used for rendering texture.

Mip Maps  →  Dithering
What Programmers Do: Use A Search Engine!

Write a utility for dithering mip map images that are used for rendering texture.

“mip map dithering texture image graphics”
What Programmers Do: Use A Search Engine!

Write a utility for dithering mip map images that are used for rendering texture.

“mip map dithering texture image graphics”

Task Description
What Programmers Do: Use A Search Engine!

Write a utility for dithering mip map images that are used for rendering texture.

“mip map dithering texture image graphics”

Task Description

Keywords
Example Source Code Search Engine

"mip map dithering texture image graphics"
Example Source Code Search Engine

Query

“mip map dithering texture image graphics”
Example Source Code Search Engine

Query

“mip map dithering texture image graphics”

List of Results
Users Prefer Web Search to Code Search

Of 35 Professional Programmers we surveyed:
• 12 did not search for code online
• 19 did not use code search engines
• 14 cited “irrelevant results” as the reason
• Preferred **Web Search to Code Search**!
Users Prefer Web Search to Code Search

Of 35 Professional Programmers we surveyed:

• 12 did not search for code online
• 19 did not use code search engines
• 14 cited “irrelevant results” as the reason
• Preferred Web Search to Code Search!

How Could This Happen?
Search Result - A Relevant Function

111 /*
112 ^ Perform variance-based color quantization on a 24-bit image.
113 */
114 int
115 txMipPal256(TxMip *pxMip, TxMip *txMip, int format, FxU32 dither, FxU32 compression)
116 {
117     int      w, h;
118     int      i;          /* Counter */
119     int      OutColors;   /* # of entries computed */
120     int      Colormax;    /* quantized full-intensity */
121     float    Cfactor;     /* Conversion factor */
122 #if 0
123     uchar   *rgbmap;     /* how to map colors to palette indices */
124 #else
125     static uchar rgbmap[(1<<NBITS)*(1<<NBITS)*(1<<NBITS)]; /* how to map colors to pa
126 #endif
127     int      pixsize;
Search Result - A Relevant Function

/*
 * Perform variance-based color quantization on a 24-bit image.
 */

int txMipPal256(TxMip *pxMip, TxMip *txMip, int format, FxU32 dither, FxU32 compression)
{
  int w, h;
  int i; /* Counter */
  int OutColors; /* # of entries computed */
  int Colormax; /* quantized full-intensity */
  float Cfactor; /* Conversion factor */

  #if 0
  uchar *rgbmap; /* how to map colors to palette indices */
  #else
  static uchar rgbmap[(1<<NBITS)*(1<<NBITS)*(1<<NBITS)]; /* how to map colors to pa
  #endif
  int pixsize;

Keywords in Comments
Find Usages of Relevant Functions

* Get a 256 color palette, to be used as samples
* Incidentally, convert src 32 bit image to dst 8 bit indexed image,
* with indices referring to the 256 color palette.
* Also incidentally, pack the alpha channel if necessary.
*/

if( txVerbose )
{
    printf("NCC Neural nets..."); fflush(stdout);
}
pxMip->format = (format == GR_TEXFMT_YIQ_422) ? GR_TEXFMT_P_8 :
    GR_TEXFMT_AP_88;
ncolors = txMipPal256(pxMip, txMip, pxMip->format, 0, 0);
if( txVerbose )
{
    printf("%d samples...", ncolors); fflush(stdout);
}
txMapPal256toYAB((FxU32 *)yabTable, (FxU8 *)map, ncolors, (FxU32 *)pxMip->pal);
if( txVerbose )
{
    printf("eMax=($3ld$3ld$3ld)...eAvg=($3ld$3ld$3ld)\n",
        errG, errR, errB,
    totG/ncolors, totR/ncolors, totB/ncolors}
Find Usages of Relevant Functions

* Get a 256 color palette, to be used as samples
* Incidentally, convert src 32 bit image to dst 8 bit indexed image,
* with indices referring to the 256 color palette.
* Also incidentally, pack the alpha channel if necessary.
*/

if ( txVerbose )
{
    printf("NCX Neural nets..."); fflush(stdout);
}
pxMip->format = (format == GR_TEXFMT_YIQ_422) ? GR_TEXFMT_P_8 :
    GR_TEXFMT_AP_88;
ncolors = txMipPal256(pxMip, txMip, pxMip->format, 0, 0);
if ( txVerbose )
{
    printf("%d samples...", ncolors); fflush(stdout);
}

if ( txMapPal256toYAB (FxU32 *)yabTable, (FxU8 *)map, ncolors, (FxU32 *)pxMip->pal);
Mag 1.94
HR 1790
1.59 \times 10^{31} \text{ kg}
Mag 1.94
HR 1790
1.59 \times 10^{31} \text{ kg}
Save()
CC 0.13

Save()

31 LOC
Our Search Engine: Portfolio

Our Contribution, Two Search Models:
1) Navigation Model for Software
2) Association Model for Software
Navigation Model
Navigation Model - PageRank

\[ PR(A) = (1-d) + d \left( \frac{PR(T_1)}{C(T_1)} + \ldots + \frac{PR(T_n)}{C(T_n)} \right) \]
Navigation Model - PageRank
Navigation Model - PageRank
Spreading Activation in Associative Networks

Spreading Activation in Associative Networks

Hawaii

Spreading Activation in Associative Networks

Hawaii

Iceland

In Portfolio: Function calls are associations

- initRenderer
- LoadTextureFromFile
- GlareTexture
- ImageTexture
- TiledTexture
- CreateTextureFromImage

Functions from Celestia: http://www.shatters.net/celestia/
In Portfolio: Function calls are associations

initRenderer

LoadTextureFromFile

GlareTexture

ImageTexture

TiledTexture

CreateTextureFromImage

Functions from Celestia: http://www.shatters.net/celestia/
Association Model – Spreading Activation

```
initRenderer

LoadTextureFromFile

GlareTexture

TexImage

TiledTexture

CreateTextureFromImage

“mip map dithering texture image graphics”

Functions from Celestia: http://www.shatters.net/celestia/
```
Association Model – Spreading Activation

Functions from Celestia: http://www.shatters.net/celestia/

initRenderer

LoadTextureFromFile

GlareTexture

ImageTexture

CreateTextureFromImage

TiledTexture

Texture image graphics

0.65
Association Model – Spreading Activation

“mip map dithering texture image graphics”

Textual Similarity 0.52

0.65

TiledTexture

CreateTextureFromImage

0.52

ImageTexture

GlareTexture

LoadTextureFromFile

initRenderer

Functions from Celestia: http://www.shatterson.net/celestia/
Association Model – Spreading Activation

“mip map dithering texture image graphics”

Textual Similarity 0.52

0.65

0.52

0.42

0.65

0.52

createTextureFromFile

loadTextureFromFile

initRenderer

imageTexture

glareTexture

TiledTexture

CreateTextureFromImage

functions from Celestia: http://www.shatters.net/celestia/
Association Model – Spreading Activation

Functions from Celestia: http://www.shatters.net/celestia/
Association Model – Spreading Activation

```
initRenderer

LoadTextureFromFile

GlareTexture

ImageTexture

TiledTexture

CreateTextureFromImage

0.27

0.33

0.52

0.65

“mip map dithering texture image graphics”

Textual Similarity

Functions from Celestia: http://www.shatters.net/celestia/
```
FCG

Function Graph Builder

Projects Archive

PageRank

Celestia

GlareTexture

ImageTexture

TiledTexture

CreateTextureFromImage

UseImage
Celestia

GlareTexture

TiledTexture

CreateTextureFromImage

ImageTexture

UseImage

Popular

FCG

Function Graph Builder

Projects Archive

PageRank

Metadata Builder

P_{PR}
FCG

Function
Graph
Builder

PageRank

Metadata
Builder

Projects
Archive

Projects
Metadata
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
/* Tiling Method for Mip Maps */
static Texture* TiledTexture(Image& img)
{
    if (GetTextureCaps().nonPow2Supported)
    {
        /* prepares mip maps for dithering */
        if (mipMode == Texture::DefaultMipMaps)
            mipMode = Texture::AutoMipMaps;
    }
    ...
}
SAN0.65
TiledTexture
Relevant Functions

SAN

FCG

Function Graph Builder

Projects Archive

PageRank

Metadata Builder

Projects Metadata

IR Engine

Celestia

SAN0.65

GlareTexture

ImageTexture

TiledTexture

CreateTextureFromImage
Software Archive

FreeBSD Ports

- 18,203 C/C++ Projects
- 2.4 Million Files
- 8.5 Million Functions
- 32 Million Function Calls
- 270 Million Total Lines of Code
Portfolio Interface

“mip map dithering texture image graphics”

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGameWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CGraphicWorld</td>
<td>ultimate</td>
</tr>
<tr>
<td>CWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CRender</td>
<td>ultimate</td>
</tr>
<tr>
<td>ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>CSound</td>
<td>ultimate</td>
</tr>
<tr>
<td>reloadConfiguration</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateProceduralTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>genBlurTextures</td>
<td>celestia</td>
</tr>
<tr>
<td>CreateTextureFromImage</td>
<td>celestia</td>
</tr>
<tr>
<td>viewOptionsMenu</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateMipMaps</td>
<td>cs-pseud</td>
</tr>
<tr>
<td>genBlurTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>BuildGaussianGlareTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>BuildGaussianDiscTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>loadTileTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>getTile</td>
<td>celestia</td>
</tr>
<tr>
<td>renderGalaxyPointSprites</td>
<td>celestia</td>
</tr>
<tr>
<td>renderSection</td>
<td>celestia</td>
</tr>
<tr>
<td>renderGlobularPointSprites</td>
<td>celestia</td>
</tr>
<tr>
<td>init</td>
<td>drift-2</td>
</tr>
<tr>
<td>resize</td>
<td>celestia</td>
</tr>
<tr>
<td>resize</td>
<td>celestia</td>
</tr>
<tr>
<td>~ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>bind</td>
<td>celestia</td>
</tr>
</tbody>
</table>

Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celeengine/render.cpp`. 
<table>
<thead>
<tr>
<th>Function Name</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGameWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CGraphicWorld</td>
<td>ultimate</td>
</tr>
<tr>
<td>CWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CRenderer</td>
<td>ultimate</td>
</tr>
<tr>
<td>ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>CSound</td>
<td>ultimate</td>
</tr>
<tr>
<td>reLoadConfiguration</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateProceduralTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>genBlurTextures</td>
<td>celestia</td>
</tr>
<tr>
<td>CreateTextureFromImage</td>
<td>celestia</td>
</tr>
<tr>
<td>viewOptionsMenu</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateMipMaps</td>
<td>cs-pseud</td>
</tr>
<tr>
<td>genBlurTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>BuildGaussianGlareTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>BuildGaussianDiscTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>loadTileTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>getTile</td>
<td>celestia</td>
</tr>
<tr>
<td>renderGalaxyPointSprites</td>
<td>celestia</td>
</tr>
<tr>
<td>renderSection</td>
<td>celestia</td>
</tr>
<tr>
<td>renderGlobularPointSprites</td>
<td>celestia</td>
</tr>
<tr>
<td>Init</td>
<td>vdrift-2</td>
</tr>
<tr>
<td>resize</td>
<td>celestia</td>
</tr>
<tr>
<td>init</td>
<td>celestia</td>
</tr>
<tr>
<td>~ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>bind</td>
<td>celestia</td>
</tr>
</tbody>
</table>

Function **genBlurTextures** is in file *ports_20k/celestia/celestia-1.6.0*.
Portfolio Interface

“mip map dithering texture image graphics”

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGameWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CGraphicWorld</td>
<td>ultimate</td>
</tr>
<tr>
<td>CWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CRenderer</td>
<td>ultimate</td>
</tr>
<tr>
<td>ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>CSound</td>
<td>ultimate</td>
</tr>
<tr>
<td>reloadConfiguration</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateProceduralTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>genBlurTextures</td>
<td>celestia</td>
</tr>
<tr>
<td>CreateTextureFromImage</td>
<td>celestia</td>
</tr>
<tr>
<td>viewOptionsMenu</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateMipMaps</td>
<td>cs-pseud</td>
</tr>
<tr>
<td>genBlurTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>BuildGaussianGlarTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>BuildGaussianDiscTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>LoadTileTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>getTile</td>
<td>celestia</td>
</tr>
<tr>
<td>renderGalaxyPointSprites</td>
<td>celestia</td>
</tr>
<tr>
<td>renderSection</td>
<td>celestia</td>
</tr>
<tr>
<td>renderGlobularPointSprites</td>
<td>celestia</td>
</tr>
<tr>
<td>init</td>
<td>drift-2</td>
</tr>
<tr>
<td>resize</td>
<td>celestia</td>
</tr>
<tr>
<td>resize</td>
<td>celestia</td>
</tr>
<tr>
<td>~ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>bind</td>
<td>celestia</td>
</tr>
</tbody>
</table>

Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp`.
Portfolio Interface

“mip map dithering texture image graphics”

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGameWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CGraphicWorld</td>
<td>ultimate</td>
</tr>
<tr>
<td>CWinSystem</td>
<td>ultimate</td>
</tr>
<tr>
<td>CRenderer</td>
<td>ultimate</td>
</tr>
<tr>
<td>ImageTexture</td>
<td>celestia</td>
</tr>
<tr>
<td>CSound</td>
<td>ultimate</td>
</tr>
<tr>
<td>reloadConfiguration</td>
<td>ultimate</td>
</tr>
<tr>
<td>CreateProceduralTexture</td>
<td></td>
</tr>
<tr>
<td>genBlurTextures</td>
<td></td>
</tr>
<tr>
<td>CreateTextureFromImage</td>
<td></td>
</tr>
<tr>
<td>viewOptionsMenu</td>
<td></td>
</tr>
<tr>
<td>CreateMipMaps</td>
<td></td>
</tr>
<tr>
<td>genBlurTexture</td>
<td></td>
</tr>
<tr>
<td>BuildGaussianGlareTexture</td>
<td></td>
</tr>
<tr>
<td>BuildGaussianDiscTexture</td>
<td></td>
</tr>
<tr>
<td>loadTileTexture</td>
<td></td>
</tr>
<tr>
<td>getTile</td>
<td></td>
</tr>
<tr>
<td>renderGalaxyPointSprites</td>
<td></td>
</tr>
<tr>
<td>renderSection</td>
<td></td>
</tr>
<tr>
<td>renderGlobularPointSprites</td>
<td></td>
</tr>
<tr>
<td>init</td>
<td></td>
</tr>
<tr>
<td>resize</td>
<td></td>
</tr>
<tr>
<td>resize_init</td>
<td></td>
</tr>
<tr>
<td>~ImageTexture</td>
<td></td>
</tr>
<tr>
<td>bind</td>
<td></td>
</tr>
</tbody>
</table>

Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celegine/render.cpp`.

Ultimate Stunts is a UNIX/Windows/Linux remake of the DOS game stunts, providing modern features like OpenGL graphics, 3D sound and Internet multiplayer. Design your own tracks, choose your opponents and try the most spectacular stunts you've ever seen.

This game is still under heavy development, don't expect everything to work.

WWW: [http://www.ultimatestunts.nl/](http://www.ultimatestunts.nl/)
Portfolio Interface

“mip map dithering texture image graphics”

Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp.`
Function `genBlurTextures` is in file `ports_20k/celestia/celestia-1.6.0/src/celengine/render.cpp`. 
Experiment

To compare **Portfolio** with **Google Code Search** and **Koders**

49 C/C++ Programmers participated
  - 44 Professionals from Accenture
  - 5 Students from the University of Illinois at Chicago
Large Case Studies are Rare

“First, it is very difficult to scale human experiments to get quantitative, significant measures of usefulness; this type of large-scale human study is very rare.

Second, comparing different recommenders using human evaluators would involve carefully designed, time-consuming experiments; this is also extremely rare.”

Saul, Filkov, Devanbu, Bird
Recommending Random Walks, ESEC/FSE’07
Participants’ Role

1) Receive Task and Search Engine
Write a utility for dithering mip map images that are used for rendering texture.

2) Translate Task to Query, enter into Engine

Portfolio
A Source Code Search Engine for FreeBSD ports

mip map dithering texture image graphics

Search

Like 7
**Likert Scale - Confidence**

🌟🌟🌟🌟 Completely irrelevant – there is absolutely nothing that the participant can use from this retrieved code fragments, nothing in it is related to keywords that the participant chose based on the descriptions of the tasks.

🌟🌟🌟🌟 Mostly irrelevant – a retrieved code fragment is only remotely relevant to a given task; it is unclear how to reuse it.

🌟🌟🌟🌟 Mostly relevant – a retrieved code fragment is relevant to a given task and participant can understand with some modest effort how to reuse it to solve a given task.

🌟🌟🌟🌟🌟 Highly relevant – The participant is highly confident that code fragment can be reused and s/he clearly see how to use it.
Analysis of the Results

Metrics:
- Confidence (C)
- Precision (P)
- Normalized Discounted Cumulative Gain (NG)

<table>
<thead>
<tr>
<th>Search Engine</th>
<th>Queries Entered</th>
<th>Responses Rated</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portfolio</td>
<td>184</td>
<td>1276</td>
</tr>
<tr>
<td>Google Code Search</td>
<td>198</td>
<td>1373</td>
</tr>
<tr>
<td>Koders</td>
<td>208</td>
<td>1486</td>
</tr>
</tbody>
</table>
Results – Confidence

<table>
<thead>
<tr>
<th></th>
<th>Google</th>
<th>Koders</th>
<th>Portfolio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image" alt="Stars" /></td>
<td><img src="image" alt="Stars" /></td>
<td><img src="image" alt="Stars" /></td>
</tr>
</tbody>
</table>

- \( p < 5 \cdot 10^{-108} \)
- \( F = 261.3 \)
- \( F_{\text{crit}} = 3.01 \)
Results – Precision

<table>
<thead>
<tr>
<th></th>
<th>p</th>
<th>F</th>
<th>F_{crit}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google</td>
<td>&lt;8.6\cdot10^{-22}</td>
<td>52.5</td>
<td>3.01</td>
</tr>
</tbody>
</table>

Precision values for Google, Koders, and Portfolio are shown in the chart.
Results – Normalized Discounted Gain

- Normalized Discounted Gain: $p < 2.5 \cdot 10^{-18}$
- $F = 43.8$
- $F_{\text{crit}} = 3.01$
Statistical Analysis – ANOVA

Null Hypothesis rejected in all cases:

$H_0$ – There is no difference in the C, P, or NG mean values among users of Portfolio, Google Code Search, and Koders.

$H_1$ – There is statistically-significant difference in the numbers of C, P, and NG mean values among users of Portfolio, Google Code Search, and Koders.

<table>
<thead>
<tr>
<th>Metric</th>
<th>p</th>
<th>F</th>
<th>$F_{critical}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence</td>
<td>&lt; 5.0 · 10^{-108}</td>
<td>261.3</td>
<td>3.01</td>
</tr>
<tr>
<td>Precision</td>
<td>&lt; 8.6 · 10^{-22}</td>
<td>52.5</td>
<td>3.01</td>
</tr>
<tr>
<td>Discounted Gain</td>
<td>&lt; 2.5 · 10^{-18}</td>
<td>43.8</td>
<td>3.01</td>
</tr>
</tbody>
</table>
Statistical Analysis – ANOVA

Null Hypothesis rejected in all cases:

\( H_0 \) – There is no difference in the C, P, or NG mean values among users of Portfolio, Google Code Search, and Koders.

\( H_1 \) – There is statistically-significant difference in the numbers of C, P, and NG mean values among users of Portfolio, Google Code Search, and Koders.

<table>
<thead>
<tr>
<th>Metric</th>
<th>p</th>
<th>F</th>
<th>F_{critical}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidence</td>
<td>&lt; 5.0 \cdot 10^{-108}</td>
<td>261.3</td>
<td>3.01</td>
</tr>
<tr>
<td>Precision</td>
<td>&lt; 8.6 \cdot 10^{-22}</td>
<td>52.5</td>
<td>3.01</td>
</tr>
<tr>
<td>Discounted Gain</td>
<td>&lt; 2.5 \cdot 10^{-18}</td>
<td>43.8</td>
<td>3.01</td>
</tr>
</tbody>
</table>
Programmer Experience Relations

Do more-experienced programmers report different results than less-experienced programmers?

Null Hypothesis **not** rejected:

\( H_0 \) – There is no difference in the C, P, or NG mean values among experienced and less-experienced users of Portfolio, Google Koder Search, and Koders.

\( H_1 \) – There is statistically-significant difference in the numbers of C, P, and NG mean values among experienced and less-experienced users of Portfolio, Google Code Search, and Koders.
Programmer Experience Relations

Do more-experienced programmers report different results than less-experienced programmers?

Null Hypothesis not rejected:

\( H_0 \) – There is no difference in the C, P, or NG mean values among experienced and less-experienced users of Portfolio, Google Koder Search, and Koders.

\( H_1 \) – There is statistically-significant difference in the numbers of C, P, and NG mean values among experienced and less-experienced users of Portfolio, Google Code Search, and Koders.
Responses from Programmers

“The search engine Portfolio is a good search tool ... developers won’t waste time exploring different projects or functions.”

“Portfolio looks into functional not based exactly on the wording like Google, so when it’s found the right function, the search is really on target.”

“The ‘code web’ of search results was very helpful for finding out which things to analyze.”
Suggestions from Programmers

“The best addition to Portfolio would be the ability to navigate through functions much like an IDE can.”

“I would like to see more feedback from the search and more options on how to search.”

“If query is misspelled, [Portfolio] does not return suggestions.”
**Ongoing Improvements**

Data Available
- All Source Code in Our Repository
- Function Dependency Extractor: FUNDEX
- Case Study Tasks and Responses

Programmer Access
- SOAP web service
- Java search now available

See [http://www.searchportfolio.net/](http://www.searchportfolio.net/)
Visit us online and at Facebook!

http://www.searchportfolio.net/